Change Propagation in Large Technical Systems

by

Monica L. Giffin

Submitted to the System Design & Management Program
In Partial Fulfillment of the Requirements for the Degree of

Master of Science in Engineering and Management

at the

Massachusetts Institute of Technology

January 2007
[February 2007]

Signature of Author _____
Monica L. Giffin
System Design & Management Program
January 16, 2007

Certified by _____
Olivier L. de Weck
Thesis Supervisor
Associate Professor of Aeronautics & Astronautics and Engineering Systems

Accepted by _____
Patrick Hale
Director, SDM Fellows Program

DISCLAIMER

MISSING PAGE(S)

70, 71, 74, 88

ABSTRACT

Propagation of engineering changes has gained increasing scrutiny as the complexity and scale of engineered systems has increased. Over the past decade academic interest has risen, yielding some small-scale in-depth studies, as well as a variety of tools aimed at aiding investigation, analysis and prediction of change propagation. This thesis applies many of the methods and seeks to apply and extend prior reasoning through examination of a large data set from industry, including data from more than 41,000 change requests (most technical, but others not) over nearly a decade. Different methods are used to analyze the data from a variety of perspectives, in both the technical and managerial realms, and the results are compared to each other and evaluated in the context of previous findings. Macro-level patterns emerge independent of smaller scale data patterns, and in many cases offer clear implications for technical management approaches for large, complex systems development.

Thesis Supervisor: Olivier de Weck
Title: Associate Professor of Aeronautics & Astronautics and Engineering Systems

ACKNOWLEDGEMENTS

Many people have contributed in a myriad of ways to the opportunity and possibility for me to take on SDM and to wrap up with a project of this scale. It has been a joy and an incredible challenge to step aside and devote myself to academic pursuits.

At MIT I would like to thank my thesis advisor, Olivier de Weck, for his unending enthusiasm and vision, and Gergana Bounova for her hard work and contributions to my understanding of this vast data set; Pat Hale, Jack Grace, Tom Allen, and all the others who work to make SDM vibrant; Nancy Leveson for her wry nuggets of wisdom, and Paul Lagace for believing in me a long time ago.

For the opportunity to use the CPM tool, I owe many thanks to Rene Keller and Claudia Eckert of the Engineering Design Centre at Cambridge University.

I would like to thank all of those from my professional life who made it possible for me to be here in the first place, with the challenges, opportunities, support and knowledge they've offered over the years. While I could never name all of those who deserve my gratitude, in particular I offer many thanks to Dave Gulla, Mark D'Valentine, John Fitzpatrick, Mike Meservey, Adam Art, and Dan Dechant and the other members of the Advanced Studies Program committee.

In addition, many, many thanks to all of my good friends, siblings, parents, and family who have been supportive throughout this tough year for all of us. You've understood and encouraged me in this endeavor even when you didn't hear from me for weeks (or months) on end, and made time for me when I came up for breath.

I could never have taken any of this on without the challenges, support, encouragement, and care of my wonderful husband, John Giffin. You've taken on so many things so that I could focus on this program and thesis. No one should have to try to be quiet all weekend, but you did try!

And, lastly, I would like to dedicate this work to the memory of my grandmother, Dolores Anderson Taylor. You live on in my heart, and in all that I do.

Table of Contents

This page intentionally left blank.

List of Figures

This page intentionally left blank.

List of Tables

Selected Nomenclature

Area	Defined segment of the project, similar to a component
Change Component	Set of interrelated change requests, as defined by parent-child and sibling relationships.
C_out(I), C_in(I)	The total count of edges originating in or terminating in Area I, where parent-child edges originate with the parent and sibling edges are bi-directional
C_{IJ}	The count of edges which originate in Area I and terminate in Area J
C_{NORM}	Normalized CPI, ratio of difference between C_out and C_in to the total C_out plus C_in, may be from -1 to 1
CPI	Change Propagation Index (see [5])
DSM	Design Structure Matrix (see [18])
ECO	Engineering Change Order
EDC	Engineering Design Centre at Cambridge University
Edge	The representation of a logical connection (in either parent-child or sibling form) between two change requests
IPT	Integrated Product Team
IT	Information Technology
Node	Representation of a change request, two nodes are connected by an edge
Pareto Principle	'The 80/20 rule', has several variations commonly referenced, these include: 80% of the result is accomplished with 20% of the effort, the first 20% of time in a program determines how 80% of the budget will be spent, 20% of your factors will give you 80% of the results
System Dynamics	An approach to modeling non-technical systems, including economies, projects, opinion, etc, through concepts including feedback loops. See [16]

1 Literature Review & Thesis Overview

This thesis seeks to build on the existing understanding of change propagation through: investigation of technical and programmatic patterns associated with propagation; application and evaluation of change propagation analysis and prediction tools currently under development; and general exploration of a data set unprecedented in the current literature. This approach will allow validation and further refinement of existing tools and approaches to understanding change propagation, its causes, surroundings, and results. The data set referenced includes more than 41,000 requested changes gathered during the development of a large technical system. These results will be examined with an eye to potential application in a program management context in addition to the traditional technical one.

1.1 Change Propagation

What is change propagation? Change propagation is the process by which a change to one part or element of an existing system configuration or design results in one or more additional changes to the system, when those changes would not have otherwise been required. Changes may propagate multiple steps or to many different areas, and the propagation may be more or less expected by those initiating the first change. Change propagation means that a simple and cost-effective change in one part of the design may have 'knock-on effects' incurring significant cost elsewhere in the system to fix problems caused by the initial change. This means that understanding change propagation in engineered systems is important in order to design, manufacture and operate those systems on schedule and within budget. While traditionally only *engineering* change propagation has been addressed, in the author's experience changes may also originate in or spread to non-engineered (non-technical) areas of the system, in some cases amplifying cost and other considerations. Therefore non-technical change requests (such as those concerning documentation) were not eliminated from the data set, and are treated equally in this thesis.

1.2 The State of the Art

When diving in to the field of change propagation for the first time, as with any field addressing a complex issue, it soon becomes apparent that while there exists a limited central body of literature the origins of the field lie in several different areas. Change propagation associated research and literature draws from information on change management (including associated processes, studies, etc), engineering design, engineering design management and product development, engineering communications, complexity theory, the need for flexibility in design, and many different methods of modeling- whether applied to areas of technical or managerial interest. In addition to the traditional awareness of, and reference to, research in the same and other fields, the interdisciplinary nature of the investigation leads to consideration of industrial contexts whenever and wherever possible. After all, the nature of the problem derives from industrial needs and goals. It is by continually increasing the complexity of designed systems under production that a heightened awareness of changes and change propagation has become necessary. Meanwhile, academic research and engineering education have nearly exclusively emphasized 'greenfield' (de novo) design, although

most products and systems are the result of modification of predecessor designs to a greater or lesser extent. More attention and energy should be devoted to such an important real world phenomenon.

While change can certainly propagate (cause other, associated changes due to the first change) when building a wagon, it probably won't be catastrophic for the maker's business. A change to one particular model of a car may be costly, however a change to a product platform for several variants of vehicles may not only be immediately prohibitive, but in the extreme may spark a series of changes in associated processes, and infrastructure. As a result, change propagation is not only a fascinating area for intellectual investigation in an academic setting, but also a high impact, bottom line problem for working engineers as our society moves farther and farther into the design, manufacture, deployment, and maintenance of complex systems.

Impacts of change propagation are clearly occurring daily, documented by the pervasive configuration management systems of industry (see [3] and [8]), but how can we approach the search for understanding and tools that this creates? Within the literature directly concerning change propagation, there are several main themes of questioning which emerge:

- Descriptions of the nature of change propagation, which state the reasons for interest and future work in the field,
- Results of studies, including descriptions of change propagation or patterns seen in small sets of data (fewer than 500 changes),
- Development of tools, with the goal of predicting change,
- Visualization (particularly as it pertains to networks of change or change propagation through a system), and
- Methods for controlling change propagation through design decisions.

The majority of the work to date within the field of change propagation has been to define and begin to characterize engineering change propagation. Eckert, Clarkson and Zanker [1] explain change propagation as follows:

> 'Engineering products are the sum of complex interactions between parts and systems. Parts have to interact with each other, with systems, and systems have to interact with other systems. As a consequence, a change to a single part or system may cause changes to other parts or systems'

In [1], they pursued this line of thought, identifying engineering changes as resulting from two main categories: initiated changes and emergent changes. Initiated changes are those intended by a stakeholder. Initiated changes occur purposefully, in order to achieve a goal or benefit (of course, the benefit may be to lessen the impact of negative system behavior, such as reducing vibration experienced by a driver, or reducing the confusion of an operator in stressing circumstances). In contrast, emergent change is unintended- when some aspect of the system design requires changing because the type of system

requires it based on earlier (and) initiated changes. While emergent properties can be positive (and that combined behavior is *why* we create large systems), the concern is for the majority of cases where the emergent behavior is negative, and the necessary attempts to correct the unexpected behavior have the possibility of being immediately required and costly.

Previous work has included a study of approximately 100 changes during a 4 month in-plant presence by Terwiesch & Loch which included interviews and other information gathering with experienced individuals involved in the change process (see [2]).

Much of the basis for this sort of investigation comes as a result of the body of configuration management knowledge and processes, with the addition of aspects of engineering management, engineering design, engineering change processes and procedures, and engineering communications.

1.3 Change Management & Configuration Management

Change management, and the accompanying discipline of configuration management, is a critical prerequisite for investigation into change propagation. Existing standards of configuration management (including DoD 5015.2 [17], which details the Department of Defense's implementation and procedural guidance for records management) are geared towards enabling culpability investigations rather than investigating propagation. While it is well understood that engineering changes occur, and that their effects can range from minimal to terminal (over-running budgets and eventually causing cancellation of the effort), a true technical analysis of the origins and aspects of changes requires more than spotty undocumented data. A configuration management process is required which documents sufficiently, yet requires little enough overhead that engineers are willing to use the system regularly and in the proscribed manner. One design alteration can look like another when reasoning is not documented- an initiated change can look much like an emergent change, unless requirements and design rationale[1] are understood in addition to the immediate set of circumstances causing the need for change to come to light. As a result, change management becomes a central topic to consider.

Wright's 1997 survey of engineering change management research [3] defined an engineering change as the modification of a component or product which has already entered production. Wright recognized that in that context the demands arising from engineering changes are seen as 'evil, foisted on the manufacturing function by design engineers who probably made a mistake in the first place.' In contrast, engineering changes may in reality be the vehicle by which the company maintains or grows market share- it is rare indeed to find a firm that does not use incremental innovation.

An important consideration in establishing change management for use in this or similar analysis is that engineering changes may indeed occur at different stages during a multi-generational product lifecycle:

---

[1] This is an entirely different area of pursuit, however intent specifications as defined by Leveson in [15] could have a significant effect in combating the lack of knowledge which exacerbates change propagation.

1. During *design* of an initial product or system: once the design of a system or product has moved beyond the purely conceptual stage, the hardware, software, requirements and other documentation associated with the product are usually put "under configuration control". This typically occurs somewhere between a System Requirements Review (SRR) and Preliminary Design Review (PDR). A formal configuration management process requires that any changes – which certainly always occur – be recorded, approved (typically by multiple levels in the organization) and implemented. In our experience the amount of change activity often increases in preparation for or as a result of major milestones in waterfall processes or as a result of learning from prototypes during iterative (spiral) design. Changes in this context will reflect the evolution of the design from conceptual design to preliminary design to detailed design. Such changes are natural and usually welcome as they will improve the quality of the product.

2. During *manufacturing* – or more generally during implementation: Once a design moves to be built, the need for new changes may arise. Some components that were used in prototypes may have to be substituted by others to enable affordable manufacturing of large quantities (volume production), errors during testing may become apparent or constraints imposed by manufacturing processes may require additional changes. Such changes are typically to be avoided as they may involve expensive capital investments such as tooling to be discarded or they may lead to unexpected production and time-to-market delays.

3. During *operations*: Changes may be needed to rectify problems that may occur during operations. Indeed, some shortcomings may only be detected once systems and products see extended periods of operational use. This may be exasperated by subjecting products and systems to operating environments that were not anticipated during design, or operators may have invented their own procedures and ways of operating that can lead to premature failures or underperformance. Such observations are often recorded by the original manufacturer and fed into concerted redesign efforts. In non-urgent cases such redesigns and changes may be applied to the next generation (or block upgrade), see also point 4. However, in serious cases retrofits and recalls may be needed. Such examples are numerous such as automotive recalls, patches to repair software bugs or retrofits to address fleet problems in commercial and military airplanes. Subtly different types of changes are those that involve upgrading a system or product after initial use.

4. For design of the *next generation*: When designing the next generation of a product or system, oftentimes the previous generation design is used as a template to which changes are applied. In some cases the changes may be very minor (e.g. automotive model "refresh") in other cases the changes may be more numerous. An interesting question often is whether to continue modifying a previous generation product or system by applying additional layers of engineering changes or whether to start de novo with a fresh design. A firm may also decide to reuse discrete modules or subsystems while discarding others. Even then changes to the interfaces between the new and old components may be needed. In all these

cases a deep understanding of engineering changes, change propagation and change impact (both technical and financial) is necessary.

Wright describes the scope of engineering change research up to 1997 as fitting into two main topical categories: 'tools' for analysis and synthesis of engineering change problems, and 'methods' to reduce manufacturing and inventory control impacts of engineering changes when made. This division hints at the lack of continuity in consideration of changes from one stage to another of a multi-generational product or platform.

Wright found that, in general, the tools associated with engineering change pertain to specific products or industries. While primarily found relating to electronic systems, there do exist some tools designed to address mechanical or combined systems. The tools often relate to documentation, although some modeling and prediction is also to be found (the most prevalent example is the use of CAD packages, which are generally held to be effective in reducing both the occurrence and effect of engineering changes), in part through up-front modeling, and in part through a design-board type communication between engineers.

On the other hand, Wright observed that the largest number of papers dealt with generally applicable methods for controlling engineering changes through manufacturing, and minimizing the effects of any necessary engineering changes. In several instances, methods researchers note that the more custom design and/or lead time involved in creating a product, the greater the detrimental effects of engineering changes when they occur.

The largest deficiency noted in the engineering change management literature as of Wright's publication was the lack of research addressing the effects of engineering change in the incremental product development process. Likewise, work was lacking in coordination with technology or organizational management disciplines, or more generally from a business process point of view. While engineering change may appear to be evil from a manufacturing perspective, it may in truth be a key asset in gradually developing a company's commercial advantage, and at the least deserves to be viewed as having 'the capacity to provide the engine for product development.' Of course, distinguishing between initiated changes, which are deliberately advancing the performance or quality of a product or system, and emergent changes, which may include rework and unplanned iterations, may be key in describing the actual relationship of engineering change to the company's situation.

Studies of engineering change management in situ (not captured by Wright's investigation) have been conducted in three Swedish engineering companies by Pikosz & Malmqvist, leading them to suggest strategies for improving change management practices [8]. Additionally, Huang & Mak [10] surveyed a significant cross section of UK manufacturing companies regarding attention to current engineering change management practices, finding that a careful balance must be struck between the effectiveness and the

efficiency of a change management system. Once recommendations such as these have been followed, and an effective change management process has been put in place, it provides the building blocks for observing change propagation- though hampered by the many un-reconciled approaches across industries, companies, products, and time.

1.4 Academic Investigation of Change Propagation

The first significant research on change propagation came in Terwiesch and Loch's study of engineering change orders for an automobile climate control system in 1999 [2]. They note the documented impacts of engineering changes in industry, and 'the growing level of parallelity in today's development processes' which will undoubtedly lead to an increasing number of changes in products underway. With engineering change related problems tending to be viewed 'more as a tragedy than as a sign of process management', they make the case that there is much to be desired in terms of attention and knowledge relating to the support process for administration of engineering change orders [ECOs].

The authors classify previous work on change management into 'Four Principles' of engineering change management:

- Avoid Unnecessary Changes,
- Reduce the Negative Impacts of an ECO,
- Detect ECOs Early, and
- Speed Up the ECO Process.

With these in mind, the focus of their study was to look at the reasons for long process lead times for engineering changes and, as a result, opportunities to speed up the whole process. They collected data on more than 100 changes from the company's information system and followed ten in greater depth with interviews and development of change case histories.

In those case histories, Terwiesch and Loch found that costs related to engineering change orders increased significantly, the later in the project that the change occurred. Among the contributing factors to extended durations to enact a change (and thus increasing associated costs) were the:

- complexity of engineering change order approval processes
- capacities of critical engineers being consumed by multiple projects with significant backlogs,
- batching of work due to large mental setup times, coordination, information release or retooling costs,
- 'snowball effect', and
- organizational issues

Most specifically relevant to change propagation is what they term the 'snowball effect', resulting from coupling, which Terwiesch and Loch classify into three groups: coupling between products and processes (where changing a product may require a process change, and vice-versa), coupling between components within a single subsystem, and coupling

between components in different subsystems. The stronger the coupling, the more likely the change is to propagate and cause associated changes, and increase the duration required to fix the original engineering change order. This description provided a springboard for deeper investigations into this particular area.

1.5 Seeking a Deeper Understanding

As change and customization became subjects of increasing study, the problems associated with changes and the associated processes came to the fore. Following a large case study at Westland Helicopters Limited, Eckert, Clarkson and Zanker [1] wrote about the situation they found there: complex, tightly connected products, in which a change to one component was very likely to propagate to other segments of the system. In harmony with Terwiesch and Loch's findings, they observed that the higher the level of connectivity between components (subsystems, systems), the more likely that one change would cause others.

After in-depth interviews with key engineers and a careful look at Westland's change process, the authors were able to come to several conclusions. First, changes were identified as resulting from initiated (intended changes from an outside source) or emergent changes (caused by the state of the design as a whole).

In this case, initiated changes often came from customers via customization requests, due to the nature of the helicopter design business. Less frequently, changes could result from the alteration of certification requirements, material or component innovations, or problems with and adjustments to prior designs.

Far more difficult to deal with were the emergent changes. Arising throughout the development process, generally breaking the surface within integration and testing steps, the root causes ranged from design to manufacturing and usage. Of course, all of this was exacerbated by the fact that the later the problem was discovered, the greater the cost to fix it. This holds in software engineering as well as hardware: even when dies and other such artifacts of manufacturing don't have to be changed, changes can require rework in integration as well as the necessity to re-run *all* of the integration and full system tests as a result of the non-continuous nature of software outputs. This can have significant cost impacts- standard wisdom in the software community (i.e. the unsubstantiated heuristic referenced often in *IEEE Software* bylines) is that testing typically consumes about a third of the development costs of a commercial software project.

Following analysis of the data, Eckert, Clarkson and Zanker came to define the nature of different components with regard to change propagation as falling into several categories, paraphrased here:

- Constants: unaffected by change, these neither absorb nor cause changes
- Absorbers: can absorb more changes than they cause
- Carriers: absorb and cause a similar number of changes
- Multipliers: generate more changes than they absorb

Additionally, in many systems there are other aspects which affect change propagation:

- Buffers: absorb some degree of change due to tolerance margins, but as changes accumulate the buffers are diminished, and eventually used up
- Resistors: segments of a system which are only changed as a last resort (these can be customer specified components, components fundamental to the whole design, or simply parts which are very expensive to change)
- Reflectors: another way to characterize resistors- any changes that come their way are 'reflected' onto other components which may change more easily.

Following these findings, the authors pursued engineering change research in other contexts, contributing greatly to building a field of study from several vantage points.

In [7], Jarratt, Eckert & Clarkson discuss the context of engineering change as an important facet of a company's ability to deliver product development efforts in a commercially viable manner (on time, within budget, etc.) Changes must be made, and changes will often propagate due to the interrelationships between parts of a product. In their examination of the process for engineering change within the Perkins Engine company, they extend earlier work in evaluating the presence of change propagation in development efforts, and how companies deal with evaluating changes through tools, methods, or other support. The study described was begun in 2002 with interviews of engineers: a description of the change process at the company was elicited, as well as examples of the effects of changes from the engineers' experience. Two of the more interesting notes were that:

> "a lot of the problems come from stupid mistakes – not from horrible ones – the big ones people think about and apply their formidable brains to it, but the little details are overlooked";
> "[paradoxically] big changes are less likely to propagate [unexpectedly] than little ones...".

While they did find that several tools were in use to support decision making, a lack of risk analysis support and a paucity of methods for capturing experience and rationale were both notable.

Four main reasons emerged as being responsible for changes propagating during the engineering change process:

- forgetfulness and/or oversight,
- lack of systems (connectivity) knowledge,
- communication breakdown or failure due to concurrent activity, and
- the emergent properties of complex systems

The relative mixture of these reasons may vary significantly- prior experience in an aerospace application showed a much higher proportion of emergent changes than in this study- but the underlying building blocks remain the same.

Overall, the authors made the case for the importance of further study of change propagation (as well as development of and models and tools to allow description and understanding of connectivity between components of a product), and advancing the knowledge baseline.

As a follow-on, Clarkson and Eckert went on to team with Simons to describe a method for predicting change propagation in [4], and offer preliminary findings of the method's efficacy based on a small set of case studies. Primarily concerned with predicting and managing changes to existing products as a result of faults or new requirements, the method makes use of Design Structure Matrices (DSMs) for indications of possible propagation paths. When coupled with measures of likelihood that changes will in fact propagate, a potential outcome is a scaled rating of the likely impact of a given change.

One criticism we have with respect to probabilities is that it is very difficult to justify the notion of a general "change probability" for a particular component without considering the larger context and the uncertainties that may be the initiators of the need to change. Indeed, some classes of changes may take entirely different paths through the system than other classes of changes, even when the component where the change initiates is the same. So, while one connected component may have a high probability of change for one class of change, it may have a very low probability of change for another class of change. For example, changes to the propulsion system (engine) of an aircraft may be require changes in the avionics if the change relates to fuel management or engine temperature monitoring, but in other cases an engine change may only affect the attachment structure and airframe integration.

While identifying several existing tools (software change propagation models, model based reasoning for design evaluation, computer aided mechanical design programs, solid modelers and product modularization approaches), the authors propose that none of the existing approaches are suited to the prediction of change propagation in large, complex systems such as a helicopter.

With all of this in mind, in [4] Clarkson, Simons and Eckert describe a method to harness past experience within an organization through sets of interviews and collaborative working sessions over a relatively brief (and thus possible) timeframe. The organizational knowledge is captured in the form of likelihood of propagation and the probable impact of propagation for each component corresponding to a part of a product model (also created from organizational knowledge).

Three specific change propagation cases were used to validate the model as constructed, comparing the actual changes and propagation patterns seen in development to the areas predicted to be the most probable recipients of change. For the three cases, the change prediction model was found to be a good indicator of actual results, although loops causing iterations were not included in the model. Unfortunately, the actual impacts of the changes were not available to compare with the predicted values.

A difficulty associated with using the knowledge gained from organizations to actually predict change's effects is that the process requires an easily understandable yet detailed enough model of the affected system. Therefore, research into the area of product modeling in support of engineering change management has been also been conducted, and is described by Jarratt, Eckert & Clarkson [13]. They state that when addressing the area of engineering change it became increasingly clear that better support is required to meet the needs of the commercial world. When the impact of a change may spread throughout or across products, a dependency upon an individual staff member to remember engineering changes (and track them mentally over extended periods) can not be commercially viable.

1.6 Modeling Systems and Propagation

The detailed understanding of a system required to use mental tracking is difficult to pass on at best, whether the potential recipients may be other experts or novice designers. While there are various tools to help record and track changes, or evaluate direct results of physical changes, there still exist no commercially available tools to help predict change propagation. However, the academic authors present their own Change Prediction Method [CPM] tool- a DSM product modeling technique used for the product models developed within the paper [6].

A key challenge, once the data has been gathered and the system has been modeled in some manner, is that of effectively visualizing change propagation. Keller, Eger, Eckert and Clarkson begin to address visualization in [6]. The complexity of the data required to assess and predict change propagation is significant, and becomes more so with added product complexity. Data is best viewed from different viewpoints, depending upon what is salient for a particular understanding to be reached or decision to be made. Understanding the characteristics of change propagation in a product requires understanding not only of the individual components of the product, but also of the direct and indirect links (whether energy, physical connection, an exchange of information, etc) between them- a significantly more difficult task, and one typically beyond the ability of a single person to maintain entirely in a mental model. Therefore, formalized visualization techniques are necessary to display change propagation data in a manner useful for those making design decisions.

The CPM tool, still in development, provides aid in depicting direct and indirect links between components, and both overall connectivity of the different component paths and predicted propagation paths, through use of DSMs, Change Risk Plots, Change Propagation Networks, and Propagation Trees. Keller, Eger, Eckert & Clarkson conclude that 'there is no "best" visualisation approach for change propagation data', and therefore advocate allowing the user to choose between a variety of different well developed visualization methods when approaching a problem will be most effective.

Following Keller, Eger, Eckert and Clarkson's initial foray, Keller, Eckert & Clarkson went on to address visualization in greater depth, with the idea of multiple viewpoints and related multiple views which can portray the appropriate information for different stakeholders. Greater consideration shows us that the complexity of a product requires

stakeholders with different perspectives (including designers, managers and customers) to have appropriate and consistent information, without being overwhelmed by the information which is available. A tool which can tailor the visible information to a stakeholder's needs and area of responsibility or interest can significantly streamline the decision making process by ensuring that the relevant information is displayed and not masked by irrelevant information around it.

Building on the concepts of fisheye views which had been identified earlier, where close (relevant) items are magnified and those further away are de-emphasized, it is proposed that it should be possible to provide a balanced mixture of global and detailed information as appropriate to a given stakeholder. Additionally, they propose approaches for validating the representations chosen through exposure to users, as a tool with poor interfaces (which fails to communicate well with the user and is difficult to use) can at worst simply be a waste of time and money. A truly effective tool should offer alternatives which appeal to a variety of users with diverse and strong preferences.

Earl, Eckert and Clarkson revisit and bring further clarity to the discussion of changes and complexity in the product design process, with further consideration of problems caused and the nature of their complexity [11]. With a focus on putting the management of change processes and analysis of change propagation into a broader context of general characteristics of change they consider the background design as the embodiment of knowledge and experience, the dynamic nature of the target (due to shifting customer requirements), and the way that change processes work not only on the implementation of the design, but on processes, resources and requirements. In this context, complexity can be considered to arise from different sources and combinations of sources. It overlays uncertain change processes and unpredictable outcomes on top of the existing designs, processes, and requirements which are typically highly ordered.

In the most general form, we are modifying previous designs to produce new or different functionality. Typically, it is actually the descriptions of products (drawings, diagrams, schema, etc) that designers interact with when determining which modifications to make. Insufficient or inaccessible descriptions may mean incorrect changes. Two main strategies are identified which are typically employed by companies trying to effectively manage engineering change: changes by a core team, and changes by a dedicated change team. The choice of method may have implications for timeliness of change versus cost.

In describing complexity, Earl, Eckert and Clarkson use the description of four main elements of design and product development laid out by Earl et al in 2005 [11]: the designer, the product, the process, and the user. They propose adding sophistication to the approach by treating each of these four elements as having both static and dynamic components- better capturing the differences between mature and innovative products. The structural complexity of the product interacts with the complexity of the events happening dynamically to that structure, leading to ever more complication. The authors posit that good descriptions which appropriately describe parts of the system or background may be used to make the complexity intellectually manageable. However, the interrelations of the descriptions themselves add yet another layer of complexity. The

descriptions must be kept current: if not updated and consistent they can provide the basis for mistakes which must be corrected with changes down the road.

1.7 Concrete Applications

With the advent of literature describing change propagation, combined with better understanding of flexibility and the idea of real options, de Weck and Suh address the area of system design with the specific goal of minimizing change propagation through preventative measures, since in previous work change propagation was described and analyzed as a phenomenon, with little mention of how to proactively address it. Rather than solely attempting to predict change propagation, one can also work to proactively shape future change propagation by methods like embedding flexibility in certain parts of the system. Flexibility facilitates future changes through a variety of strategies:

- turning some components from multipliers into absorbers,
- adding buffer components (effectively absorbers) into a change propagation path,
- making some components cheaper or easier to change (e.g. by implementing them in software rather then hardware), or
- splitting large monolithic components into smaller components.

In [5], they describe and address the problem of change propagation as it relates to automobile design and manufacturing, particularly in the case of developing product platforms with the flexibility to evolve over time. Most importantly, different classes of changes (e.g. length changes of an automotive chassis, changes to the upper body for styling reasons) will be triggered by particular exogenous uncertainties. By modeling the propagation of changes in functional requirements to system variable changes and ultimately to physical components, those parts of the system can be identified that could benefit from flexibility.

Product platforms (see Martin & Ishii [10] for more detail) are created with the goal of providing a base design or functionality atop which changes can be layered in designated modules in order to create a family of related, but distinct, variants. The goal of a product platform is to address a greater range of market needs whether at one time or throughout the evolution of the product line. By segmenting the product platform carefully, de Weck and Suh found that specific parts of the design can be isolated as being likely future change multipliers and turned in to change absorbers a priori, and effort can be directed towards ensuring that manufacturing methods are chosen with flexibility to support future changes. The greater the embedded flexibility in the original product platform design, the lower the cost of changes (in switching costs) later on and, most likely, through the life of the product line (despite the greater initial tooling and equipment costs.)

However, flexibility may only be useful if changes actually are needed in those components in which flexibility has been embedded. Another important finding of the work was that the classification into change multipliers, carriers, absorbers and constants by itself is insufficient to fully understand change propagation. A component may be a change multiplier (more changes radiating out than coming in), but if the component

itself (and its connected components that also need to be changed) is inexpensive to change, one may not have to be overly concerned. Conversely, a component that is classified as a carrier (same number of changes coming in as going out) could be very expensive to change – more expensive in fact than a multiplier – and therefore the notion of change cost, or switching cost, must also be present to support decision making.

1.8 Putting it All Together

How does one approach such a varied field? In order to obtain a better understanding of the different areas of research and references which have influenced the core of the change propagation literature, while I was investigating the literature cited above I started mapping the citations within each paper (starting with de Weck & Suh [5]) by the general bent of the citation (e.g. work on modeling, information on engineering change processes, general research context), and then followed the citations specifically for change propagation (or for papers very commonly cited) on to other papers. The writings cited by [5], and [7-12] were mapped by the nature of the citation and then grouped into cohesive areas. All but one of the resulting areas are mapped within the diagram below.

Those papers cited specifically for change propagation are within the center oval, while papers cited for more than one aspect are shown within overlapping areas. Papers cited the most heavily are closer to the dark oval (although the majority of the works were only cited once). Management related and technical related areas are present in roughly equal proportion.

The only area identified but not mapped on this diagram is citations of commercial context- generally referring to accelerating product cycles, segmented markets, and cost pressures. These are relevant to engineering and current business concerns in general, but less critical to understanding the interactions involved in change propagation as a field.

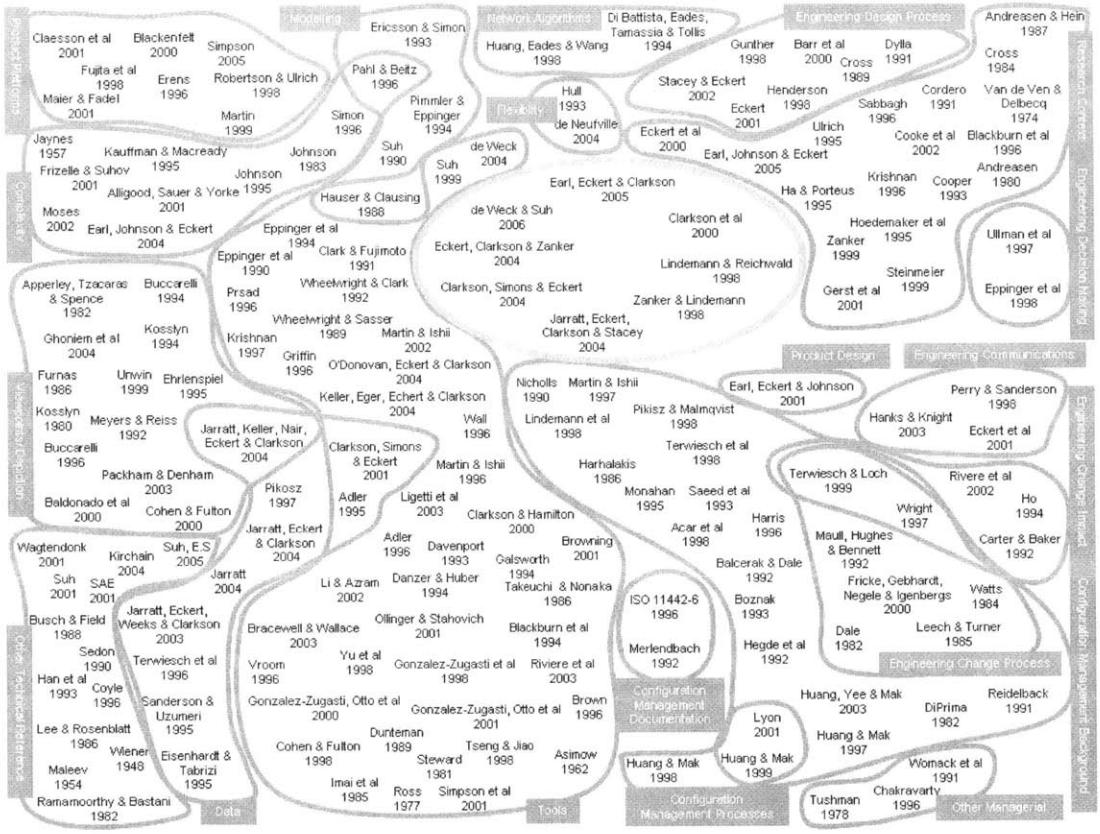


Figure 1: Change Propagation Citations

While this diagram is probably incomplete, it serves to give a picture (and reminder) of the overall context of change propagation. It concerns engineers and managers, regulators and designers. It edges into the fields of complexity, flexibility and network algorithms, as well as engineering communications and general managerial concerns. Changes necessitate give-and-take between stakeholders as well as between components. All of these interactions, which we are only beginning to understand, make for a complex and fascinating subject of academic pursuit.

This thesis was conceived based on the idea that perhaps we could use the tremendous amounts of data from one real life example to gain a better foothold in the quest to understand large scale technical projects being undertaken. While the specifics of the system cannot be given in detail here, it shall suffice to say that the system can be roughly classified as a sensor system with high complexity of hardware, software and user interactions. The data set is that of a complex, large technical program during development with multiple customers, more stakeholders, and shifting goals, spread over nearly a decade. As such, we are interested to see what patterns it may yield, and what it might tell us about the directions we are looking.

While examples of actual change analysis have been published [e.g. 1, 2], these examples typically concern only small samples of changes (<100 change requests) and therefore represent an incomplete record and the examples do not capture the full complexity of change activity on large scale projects. There are a number of reasons why previous work on data mining in terms of large scale engineering changes is lacking:

- Many firms are only interested in tracking changes while projects are ongoing. Once changes are either completed or rejected they become part of a historical record which is rarely examined as the firm moves on to the next project.

- As mentioned above changes are often due to oversights or engineering errors and there may be no incentive in exposing flawed or less than ideal processes to a wider audience. Proactive firms, however, might view a deeper understanding of change management and propagation on past projects as an opportunity for learning and continuous improvement.

- The data set may be very large, saved in heterogonous formats (e.g. due to switchover of IT systems during the course of a project), and generally difficult to access, mine, analyze and visualize.

This thesis provides a unique opportunity for examining the change records of a large project, where changes were recorded consistently over a period of nearly 9 years, with the author having intimate knowledge of the system development and change management over more than half of that time period.

1.9 Objectives

In the contract-based (B2B or B2G) segment of the commercial/industrial world (in contrast to B2C consumer markets), it is well understood that changes will be needed as a

product is developed and a program moves forward. The contracting entity may have shifting needs or budgetary constraints, and once-good assumptions may be invalidated. These changes must be dealt with as they emerge, and often on a rapid basis. As shown in the next chapter, during one particular month covered by this data, more than 1,300 change requests were generated, with an average through the program of almost 450 changes per month. This requires configuration management and change management tactics, and an understanding of implications of decisions both on a technical and a managerial basis.

This leads to an additional area to be addressed from the view of change propagation: are there managerial decisions (addressing composition of teams, program processes, etc) which can help to control change propagation? The author's industry experience has been crucial in understanding and illuminating the many contributing factors to the patterns in the data set. Therefore, this thesis seeks to evaluate this set of data from a combined industrial, commercial and academic viewpoint.

Building on the existing understanding of change propagation as it has been described in this chapter, the investigation of both technical and programmatic patterns associated with propagation will begin with Chapter 2, along with a description of the data set in detail, as well as general data processing procedures and research approach. In Chapter 3 the data patterns revealed by the processing in Chapter 2 will be described and evaluated for relevance and concurrence or contradiction with prior work and tools. In Chapter 4, potential applications of the results from different aspects of the data set will be explored, in both technical and managerial contexts. Chapter 5 will summarize the results, and address potential areas for further research, highlighting the benefits which could be gained through further partnerships for both the academic and commercial worlds.

2. Research Approach

Data for this thesis is the result of a large technical program. The program was performed on a government contractual basis, with multiple stakeholders, and involved complex hardware and software subsystems and interactions, as well as distributed users and operators, and can be broadly described as a sensor system.

What sets this study apart from other published research in change management and change propagation is that it is based on a rich data set of actual changes.

Over the course of the documented program history in the data set, several hundred individuals are referenced in the change documentation alone. These individuals included the development company's personnel, subcontracting personnel, prime contractor personnel, and customer representatives. The majority of those named were software or systems engineers, but many represent other areas of expertise.

The different groups represented are indicative of the complexities resulting from multiple customers with sometimes conflicting requirements, tensions between technical requirements, and shifting managerial focus. Different phases of the program focused on contributions of very different groups (for instance software detailed design versus system test), and often had different dominant sources of change request origin and prioritization.

The data set evaluated in this thesis consists of more than 41,000 proposed changes- technical, managerial, and procedural- over the course of nine calendar years on a single large technical program. In contrast to some previous work, the definition of change is used to refer to any changes made after initial design and implementation (this is more in keeping with Cohen & Fulton's view of engineering changes [14], and expanded to include the non-technical changes which often cause or accompany the traditional 'engineering changes'). In later portions of the data the character becomes more similar to the definition used by Wright [3] (a version of the product in use by customers is changed and the new version is released back to the customers), however we can see that the same considerations are driving decision making, and in some cases the effects can be more severe (if only internally visible) due to the relative fluidity of a product in the earlier stages of development.

The data referenced here was saved in a company developed configuration management system, and individual records contained different subsets of information (based on the programmatic guidelines for recording that information, availability of information, and the personal adherence to the stated guidelines by those entering information). The data was extracted from the configuration management system for this thesis, and processed as described later in this chapter.

2.1 The Program

A priori, the program could be broken into a few primary areas: software, hardware, and documentation. Software accounted for the vast majority of the undertaking, therefore we

would expect changes to the software to be the most prominent set of records in the data set, followed by changes to software related documentation (including requirements, design, training, and operating documentation). The hardware segment of the program required very limited development of new components- the vast majority would be reused from a pre-existing system due to the careful introduction of a buffer component between the reused hardware design and the new software.

Within the software existed a number of subsystems and components. These were logically distributed in the design based on corporate experience with similar programs, generally being differentiated based on function.

The system map below was derived from the detailed design phase artifacts describing the various subsystems and designed interactions. Interactions mainly consisted of information / data transfer.

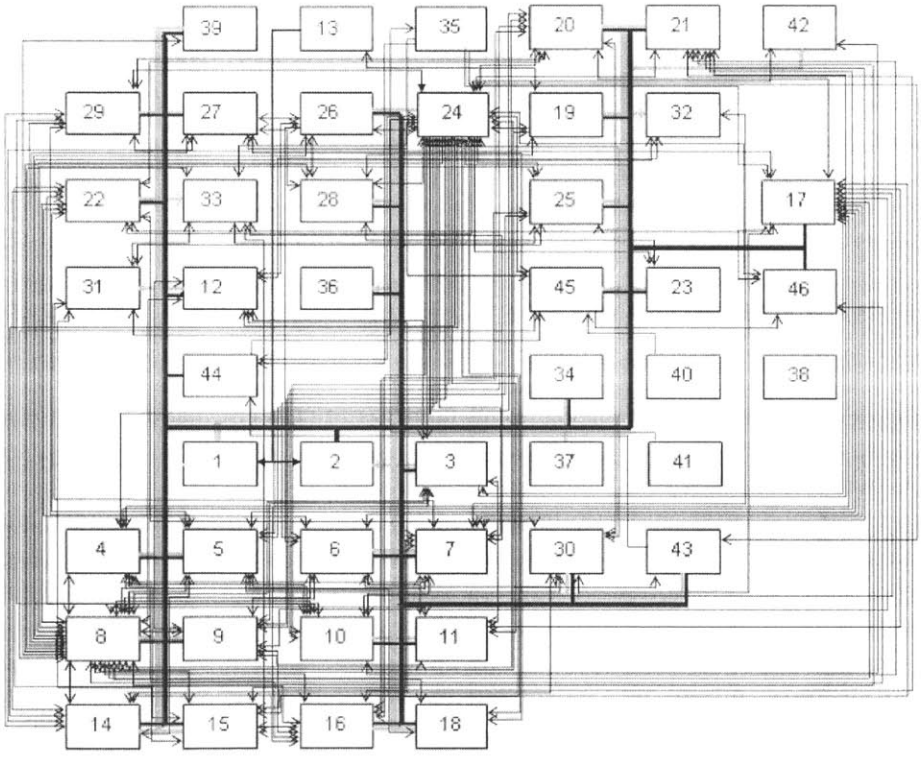


Figure 2: System Map

An initial evaluation of the detailed design data identified 46 'areas' with the potential to be affected by proposed changes. These include software components, different levels and types of documentation, and hardware. In addition, some change proposals could not be readily associated with an area. The term 'area' will be used in this context throughout this paper, and may be thought of as a coherent segment, perhaps analogous to a

subsystem. The official (structural) relationships of the areas are documented in the system map, but there are other methods available to convey the information. The figure below is another depiction of the area network, in this case from the CPM tool [6], with a single area and all of its connections to other areas highlighted in gray. A third method, Design Structure Matrices [DSMs], will be described later in this chapter, and a structural DSM representation will be used in much of the data analysis in this thesis.

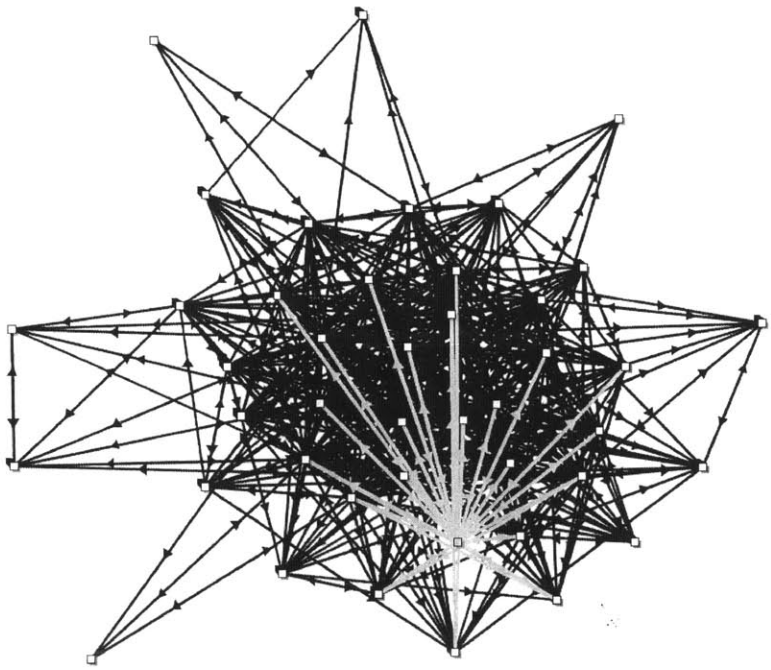


Figure 3: Area Network Depiction Highlighting Area 13 from the CPM Tool

2.2 Useful Aspects of the Data Set

A noteworthy aspect of this data set, in addition to its size, is the method of change tracking which was used. Contrary to common practice in the industry, all changes resulting from a single problem were not tracked under the same identifier. Instead, unique identifiers were used for changes on a per-area basis. Thus, if a customer complaint identified a problem with the system, and fixing that problem required changes in three different areas, the process called for three unique identifiers to be used, with acknowledgement of association (typically through parent-child or sibling relationships, although actual practice often substituted textual notes for formally noting the relationships in the configuration management tool).

In addition to documenting relationships, a wealth of information is encoded in the data set regarding impact, timing, and decision making. This information allows some insight into the commercial effects and managerial challenges resulting from these changes. During the period of time documented in this data set there were changes in program management, changes in direction to the different engineering specialties, changes in the customer representatives, and changes in overall stated program goals as they related to

other programs and strategic considerations. Each of these changes brought with it differences in how the reports which constitute this data were filled out, how they were evaluated, what sort of timeframe they would be processed in, how they were prioritized, and how they would be allowed to be implemented. Changes in Area 35 of the data were related to general program managerial concerns- typically establishing or changing programmatic processes. Other impacts are less obvious- changes in program management brought new ways of working across functions, new interfaces with the customers, and new priorities, most of which were phased in over time.

The shifts in overall strategic importance and positioning of the program initiated a different level of customer attention, eventually resulting in direct customer feedback starting during the late stages of development (this was not by its nature a 'spiral' or 'rapid prototyping' project- rather it followed a 'stage gate' process). This customer feedback came in more or less official contexts, but in the vast majority of cases fed almost directly into one or more change requests (these cases are noted in the data)- yet another rich facet of the data set.

2.3 Extraction Methodology

In order to obtain a manageable data set from the original change requests archived in the change management database, several processes were used.

First, a subset of each change request was written to a text file. This allowed a slight initial narrowing of the amount of data to be manipulated. Items such as specific build identifiers for software changes, and location codes (for changes submitted from a physical location other than the prime development facility) were omitted here.

Next, the text file thus generated was parsed and written into a MySQL database using a Perl script (see Appendix A for general examples of the Perl scripts used during this analysis). Each data member present in the text file was preserved across this transition, with all of the change numbers entered into a column, any open dates in another, etc.

With all of the data in a manageable format (the text file would have been upwards of 120,000 pages if pulled into a Microsoft Word document), the task of sorting, categorizing, anonymizing and filtering data became possible.

2.4 Anonymization

The data entries were anonymized to allow for public release by altering the following:

- names of individuals
- company and place names,
- program names, and
- dates.

The names of 496 individuals found to be mentioned in the entries were converted to a numbered Engineer or Administrator identifier. This conversion included nicknames, usernames, and the like via a set of manual SQL queries. This preserved the linking of individuals to change requests (in fact making that task far more plausible to automate)

while reserving identities. Additionally, the relative nature of the dates was preserved- a change occuring on 15-JAN-Y2 took place one year before a change occuring on 15-JAN-Y3.

2.5 Reassembly for Analysis & Tool Use

Following the considerable effort of disguising the data, a simplified change record format was written out via another Perl script. Each data entry preserves the original change request identifier, and is printed in the following format (a theoretical example, with a smaller amount of information missing than typical, is shown in the right hand column):

Table 1: Example Change Request Format

ID Number	12345
Date Created Date Last Updated	06-MAR-Y5 10-JAN-Y6
Area Affected	19
Change Magnitude	3
Parent ID	8648
Children ID(s)	15678, 16789
Sibling ID(s)	9728
Submitter	eng231
Assignees	eng008 eng231 eng018 eng018 eng018 eng271 eng231 eng008
Associated Individuals	Engineer_231 Engineer_008 Engineer_018 Engineer_018 Engineer_018 Engineer_028 Admin_001 Engineer_271 Admin_001 Engineer_028 Admin_006 Engineer_064
Stage Originated, Defect Reason	[blank], [blank]
Severity	[blank]
Completed?	1

- **ID Numbers** were assigned in a purely chronological order based upon when the change request was first submitted to the electronic database (Date Created). A small set of ID numbers are not included for various reasons, however 41,551 entries are present with the lowest ID of 1 and the highest ID of 41,594.

- **Date Created** notes the time at which the change request was entered into the change management system. **Date Last Updated** is a function of that system as well, noting the last time anything was altered for that change request.

- **Area Affected** describes the segment of the system affected. As described above, 46 'Areas' were identified (Figure 2: System Map), including requirements documentation, individual functional areas of the system software, system procedural documentation, and others.

- **Change Magnitude** is a categorization of the anticipated impact of the request. Total Hours required (non-recurring engineering effort) or Total SLOC (Source

Lines of Code) affected were recorded in many cases (although these values apply only to implementation and initial testing, and do not contain integration or subsequent testing). Magnitudes were recorded as 0 to 5 [or -1 if no information was present.] The magnitude assigned was based on the logic regularly used by the program management team when assessing potential impacts and risks of changes. Change magnitude was set based on Total SLOC if applicable, otherwise Total Hours, as show by Table 1 below:

Table 2: Change Magnitude Classification

5	Total SLOC $\geq$ 1000	$\geq$ 200 Total Hours
4	200 $\leq$ Total SLOC < 1000	80 $\leq$ Total Hours < 200
3	50 $\leq$ Total SLOC < 200	40 $\leq$ Total Hours < 80
2	10 $\leq$ Total SLOC < 50	8 $\leq$ Total Hours < 40
1	1 $\leq$ Total SLOC < 10	1 $\leq$ Total Hours < 8
0	Total SLOC < 1	Total Hours < 1

Based on general patterns of data sets, a priori we would expect changes with magnitude of 0, 1, or 2 to be the vast majority of those in the data set, with a moderate number of category 3 changes, and very few of category 4 or 5. When considering whether the changes requested would be approved and completed it is less clear what to expect: we would expect that those which could not be evaluated would have been disapproved at higher rates, while those changes to which significant resources had already been committed prior to consideration would be incorporated. Also, in some cases changes might have been rejected because the affected area was acting as a change "reflector", i.e. an area that was deemed too tightly integrated or too risky to change, see [1] for a description of reflectors in the context of a helicopter. In those cases the change disapproval might be followed by initiation of a related change request in a different area with the intent of achieving the same functional effect that was intended by the original change request.

- The **Parent**, **Children**, and **Sibling** fields identify other associated change request records by ID number. A child is a direct result of a parent (typically the parent is the original problem description, or a requirements change, while the child is one of a set of changes required to correct the problem or implement the enhancement stated in the parent). Siblings may either be children of the same parent, or otherwise related (this was evaluated based on mention of one change request by ID number in the text of another). Siblings may often supersede one another if in the same area, or may be complementary changes in different areas required to realize a common goal. The figure below is a graphical description of some potential relationships of change requests. Parent-child relationships are shown as solid, unidirectional arrows, while sibling relationships are shown as dashed, bidirectional arrows.

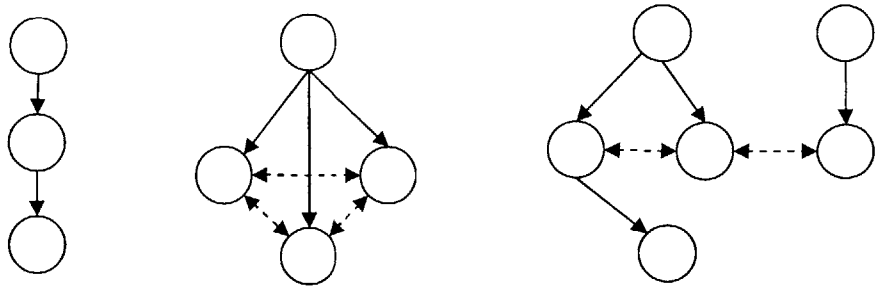


Figure 4: Example Change Request Relationships

- Individuals who were instrumental in processing of a change request are indicated by the **Submitter**, **Assignees**, and **Associated Individuals** fields. Submitters entered the change request into the database. Assignees formally possessed responsibility for the change request at some point (although the decision to change assignees could occur either formally- through evaluation at a review board- or informally through discussion between administrators and/or engineers). Associated Individuals are those mentioned by name in the textual descriptions within the change request. The three fields are not mutually exclusive, so a given engineer may be identified in all three for a given change request (and in fact may occur multiple times in the same request's Assignees or Associated Individuals field due to being reassigned to the change request or bring referenced multiple times in the text).

- **Stage Originated**, **Defect Reason**, and **Severity** are all fields from the original change request, although Stage Originated and Defect Reason also contain consolidated data from another field portraying whether a change request was made as a direct result of a documented customer request (a change request initiated from outside the project). These fields are sparsely populated through the data set, yet may yield some insight, particularly where the customer requests are concerned.

- The last field, '**Completed?**', is a numeric indicator of the final status of the change request. A value of 1 indicates that the original purpose of the change request was completed (be it investigation of design details, incorporation of suggested document changes, hardware interface definition, etc). A value of 0 indicates that the entry had a status indicating that it was still in process as of data set capture (whether under further investigation, deferred for the time being, or waiting for the results to be officially incorporated). A value of -1 indicates that the change request was withdrawn, disapproved or superseded by another change request. If a status value had not been entered into the appropriate field, a null value was automatically assigned in the first pass. In some cases further textual examination allowed evaluation, however it was not within the bounds of possibility for this effort to do so for every entry.

2.6 Data Analysis Processing

The data set described above was evaluated by a combination of methods. The anonymized records were input to a Matlab network tool created by Gergana Bounova, which allowed evaluation of the edges (connections between individual change request nodes), as well as identification of change components (sets of mutually linked change requests). Other data collation and processing was done using a mix of SQL database queries, Perl scripts, and Matlab and Excel manipulation. Additionally, the area change network information from the Matlab tool was manipulated into an XML model file for use in the CPM tool created by the University of Cambridge Engineering Design Centre (EDC) group and described in [4] and [12], among others.

A more detailed description of the data extraction results and initial manipulation follows.

2.7 Summary of Data Set Characteristics

Of the 41,551 change request records in the data set, change magnitudes could not be determined for 15,465. The 26,086 which could be evaluated from the provided data were distributed as follows:

Table 3: Status by Magnitude (Initial)

	Total Number	Completed	Unresolved / In Process	Withdrawn / Superseded / Disapproved
5	124	120 (96.7%)	0	4 (3.3%)
4	751	724 (96.4%)	3 (0.4%)	24 (3.2%)
3	2,048	1942 (94.8%)	11 (0.5%)	95 (4.6%)
2	5,296	4623 (87.3%)	13 (0.2%)	223 (4.2%)
1	7,430	6937 (93.3%)	58 (0.8%)	435 (5.9%)
0	10,437	5323 (51%)	255 (2.4%)	4859 (46.6%)

For those cases where the change magnitude could be determined, 437 of those rated a magnitude of '2' could not automatically have their completion status be determined due to failure to adhere to change management processes, thus the discrepancy in the numbers for the magnitude 2 row of the table. After manual evaluation of completion (based on the textual narration in the change requests), we see the results displayed in the table and figure below.

Table 4: Status by Magnitude (Revised)

	Total Number	Completed	Unresolved / In Process	Withdrawn / Superseded / Disapproved
5	124	120 (96.7%)	0	4 (3.3%)
4	751	724 (96.4%)	3 (0.4%)	24 (3.2%)
3	2,049	1942 (94.8%)	11 (0.5%)	96 (4.7%)
2	5,295	4918 (92.8%)	14 (0.3%)	363 (6.9%)
1	7,430	6937 (93.3%)	58 (0.8%)	435 (5.9%)
0	10,437	5323 (51%)	255 (2.4%)	4859 (46.6%)

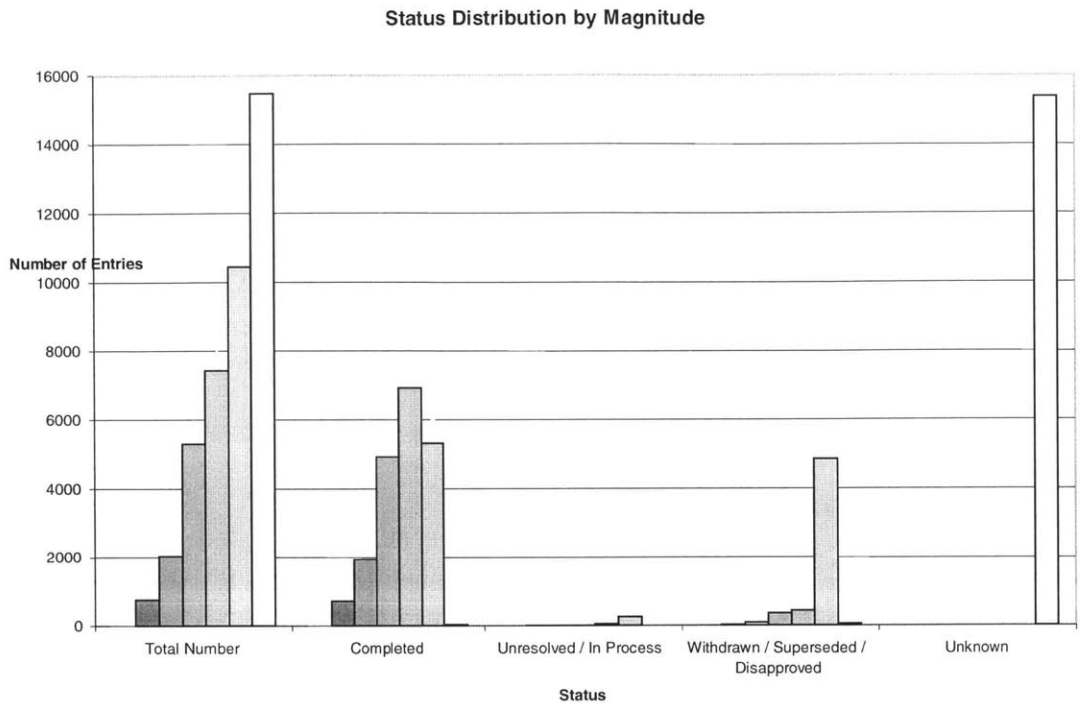


Figure 5: Status Distribution by Magnitude

It should be noted that during the investigation of the 437 initially undetermined magnitude 2 change requests, one was identified to be magnitude 3: the impact values had also not been recorded in compliance with the data collection process. It is expected that other entries have similar errors, however it is beyond the scope of this effort to verify consistency of every individual data record.

Despite some of the noise in the underlying data there are two key observations that can be made:

1. There is a monotonic (nearly exponential) relationship between expected magnitude of change and its frequency of occurrence. Very large changes are rather infrequent, small changes on the other hand are commonplace. The hypothesis is that this distribution follows the Pareto principle (80/20 law)

2. Nearly half the small (magnitude 0) changes were either withdrawn, superseded or disapproved (46.6%), whereas nearly all the large changes (magnitude 5) were approved and carried out to completion (96.7%). The reason for this is not immediately clear, although it may be related to how quickly effort required increases for the large changes, even prior to much of the evaluation process.

On further reflection, the discovery of an obvious inconsistency in this area should not be unexpected, given that this set of entries was identified for examination due to lack of adherence to change management processes in the first place. Of the 15,465 records for which change magnitude could not be determined, 15,368 were also lacking status data in

the required format. While these entries must be omitted from consideration of propagation impact values, they can be taken as an approximation of process non-adherence over the course of the program.

Additionally, of the non-compliant records, 130 contained no data whatsoever, and do not factor into any of the other calculations in this thesis. Likewise, area 34, while expected to have changes, had none associated in the data set. Figure 4 shows the relative distribution of completed, unresolved, withdrawn/superseded and unknown-status change requests by area, and while in some areas the completion rate is near 80% (e.g. areas 1, 14, 38,42), in other areas it is below 20% (areas 2, 11, 18, 26, 28, 33, 35)- this indicates potentially interesting differences in the areas that bears further investigation.

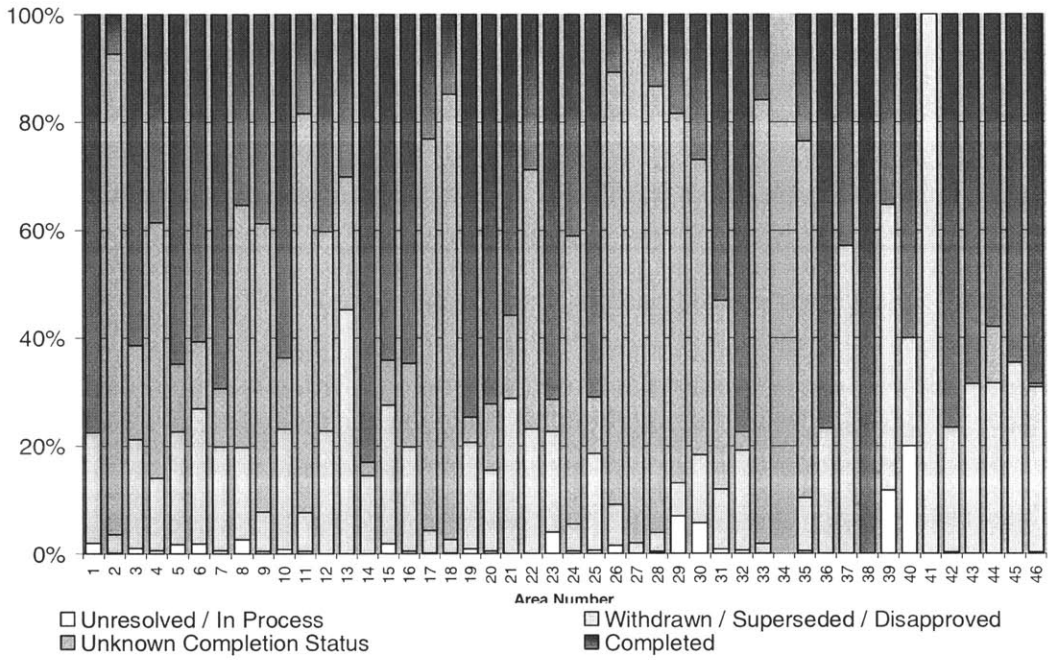


Figure 6: Status (in %) By Area

The graph below, of the non compliant change records over time at a macro level, indicates a general positive trend in adherence to established change management

procedures over the course of the program.

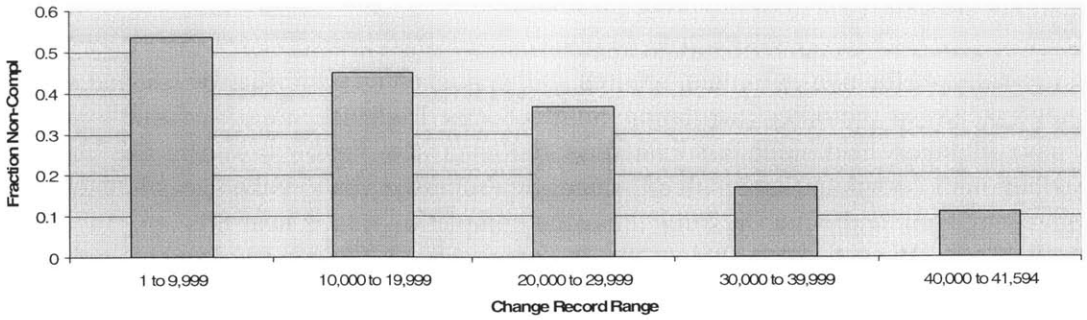


Figure 7: Fraction Non-Compliant Records in Major Time Blocks

A detailed look at the non-compliance, categorizing the records by years into the program at creation (first 12 months, second, etc), reveals a significant increase in the fraction of non-compliant change records generated in the last year of program data.

There are many possible explanations, but the most likely is a combination of two causes. First, the increase is probably in part due to the extended duration of some of the program processes (which may take several months to close out a record in some cases, and place a smaller premium on completing paperwork than other tasks), and secondly in part due to personnel changeover due to transition to a final sell-off effort phase.

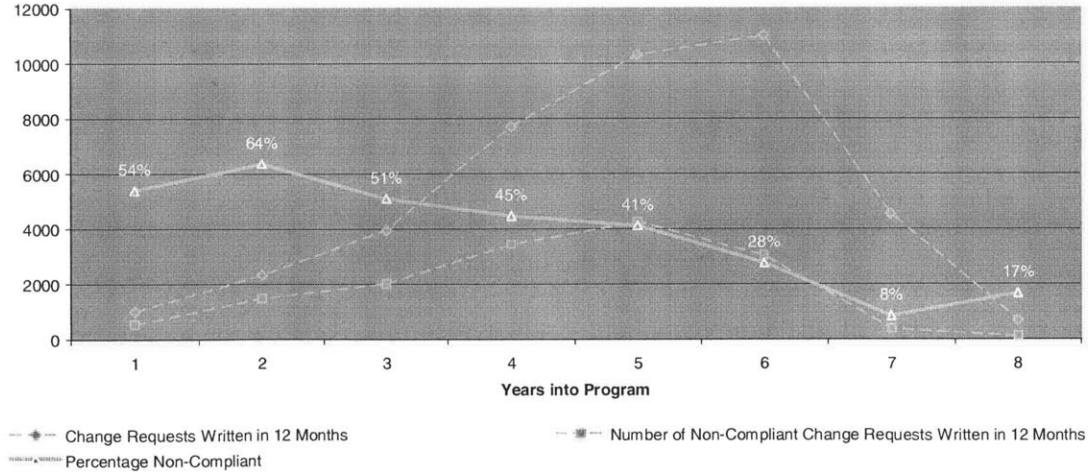


Figure 8: Yearly Change Request Generation

Potential side effects from increased process adherence would logically include both

- Less unanticipated change propagation, due to more rigorous evaluation of potential changes, and

- Slightly increased impact values for changes (particularly regarding documentation), reflecting increased investigatory and unit testing overhead

However, while likely present, these effects could not be discerned from others without significant further data processing which is beyond the scope of this thesis.

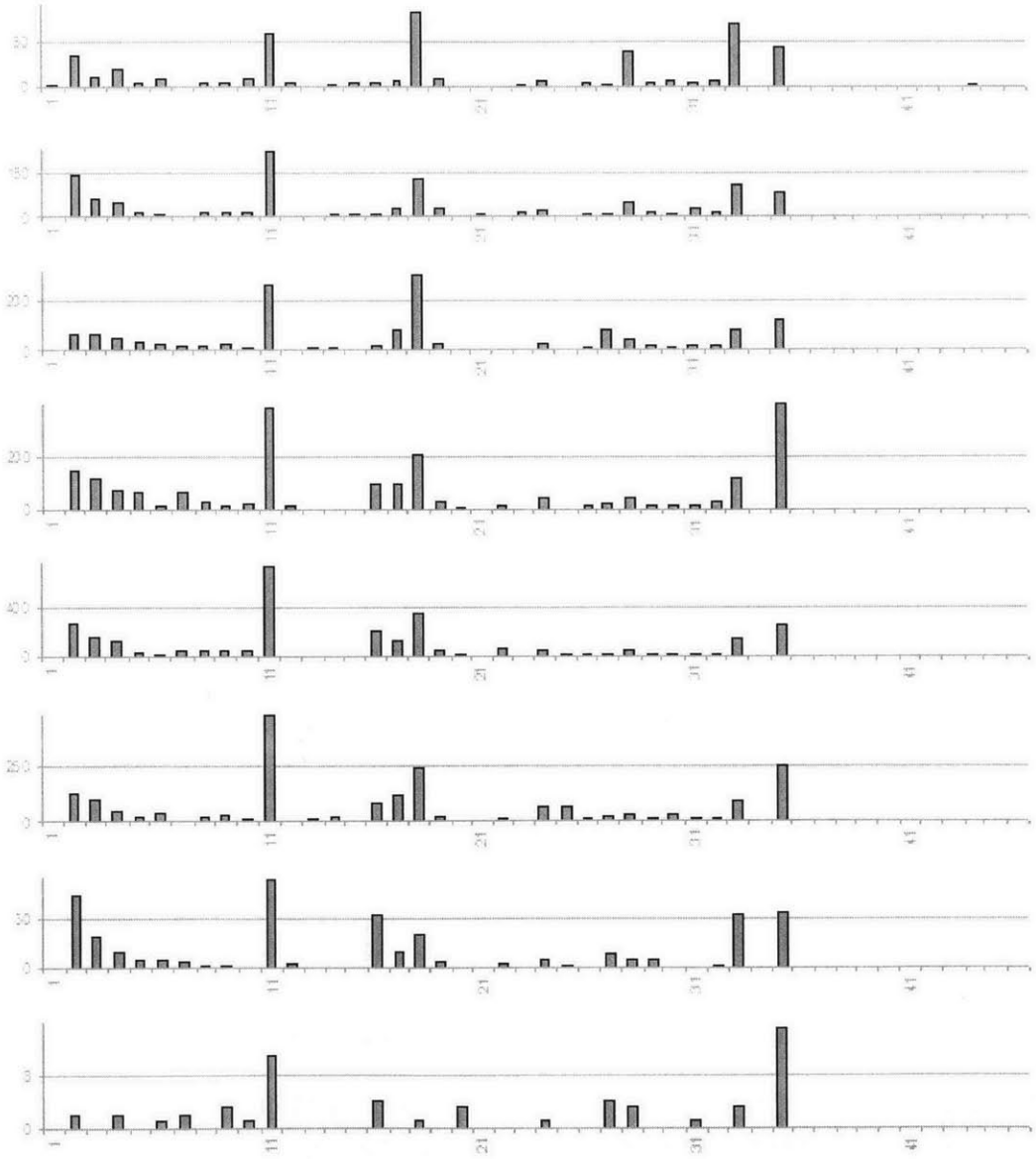


Figure 9: Eight Years of Non-Compliance, By Area

Interestingly, a quick glance at the sparklines (Figure 9) showing non-compliance by area for each of the years in the program at least indicates that the same areas were contributing to non-compliance year after year, and give an indication of which areas are continually disregarding process (and therefore might see a larger number of associated

propagations, assuming process is a good minimizer of propagation as a result of instituting exploration of potential impacts prior to implementation of changes).

It is interesting to note that areas 11 and 35 (despite being functionally very different from each other) appear to consistently have a significant number of non-compliant change requests in comparison to the other areas, indicating yet another potential avenue for future investigation (for instance, one potential explanation is a lack of process discipline on the part of individuals in charge of those areas).

While time-based evaluations of non-compliance are fascinating, further exploration of the entire data set by time period is worthwhile, and instead we can break the whole data set down into total change request generation on a monthly basis.

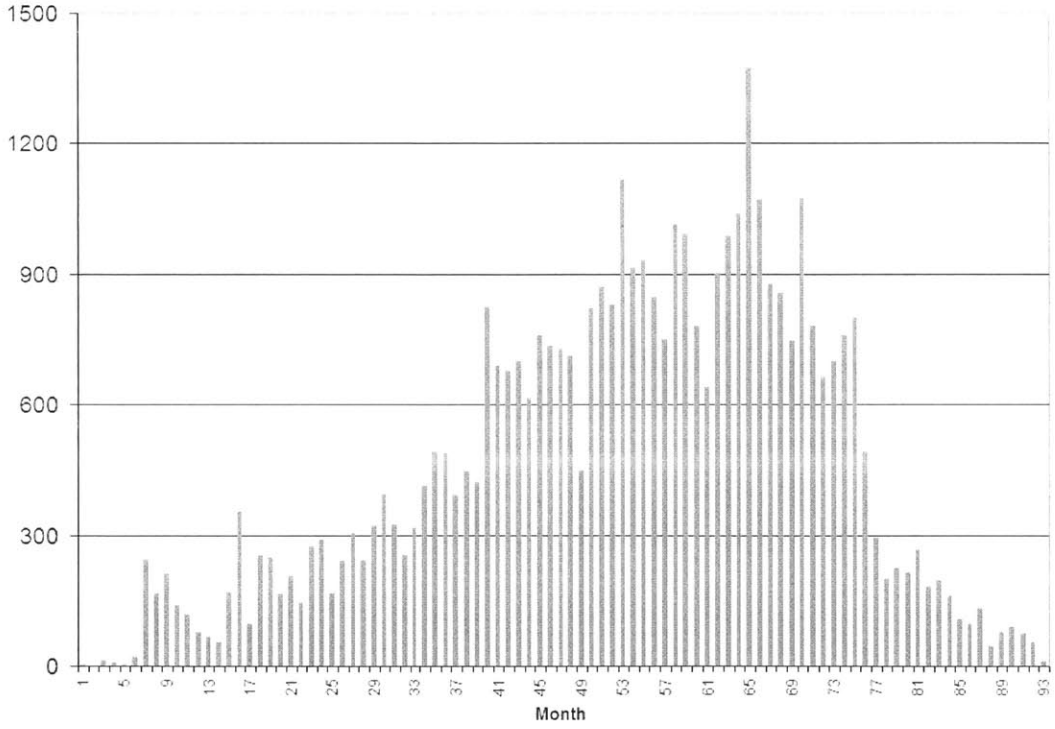


Figure 10: Monthly Change Request Generation

At this level we see the emergence of possible periodicity and interesting fluctuations. This will be discussed in more detail in Chapter 3

2.8 Building Blocks for Analysis

While the above discussion addresses the fact that change was in fact happening and tracked at an aggregate level, it is necessary to explore how those changes associated with the system affected different areas. Did they occur in expected areas? Were they uniformly distributed throughout the system? Either way, can anything useful be discerned from those patterns?

In order to better understand to what extent changes propagating through the system followed expected connections, the expectations must be mapped into what will be referred to here as the *structural DSM* (Design Structure Matrix, see [18]). This format captures the expected flow of changes between 'adjacent' areas of the program as they were defined in the detailed design information (a consolidated version of which was shown in the system map, Figure 2), which may be of several types. Potential examples include change passing from customer originating documents to the documents governing processes used in design reviews, from coding standards to code, from one software or hardware component to another, and from a component to operations documents. For the sake of simplicity, the structural DSM simply codes these all as connections from one area (a column) to another area (a row). This offers a much more accessible form of the information provided in Figure 2.

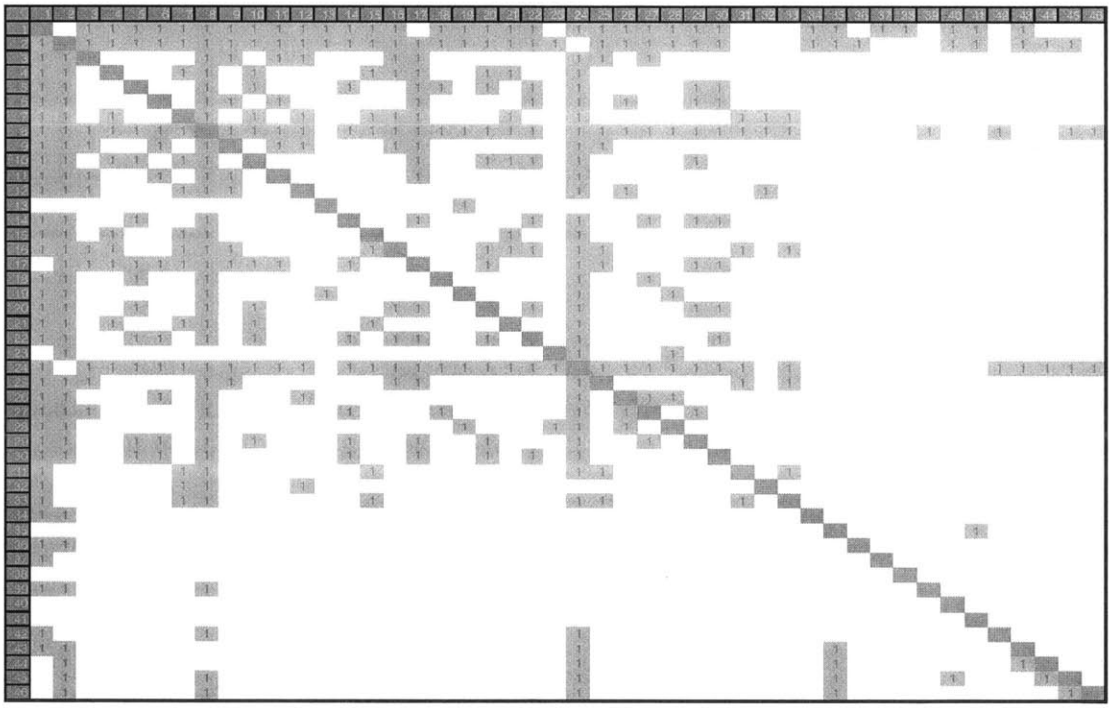


Figure 11: Unclustered Structural DSM

While the structural DSM is largely symmetric, it is quite logically not entirely so if we think generally about the issue of information flow (and portraying a whole program's interrelationships in this manner). One example would be legislative guidelines- they can directly affect requirements documents, but the opposite is not true. The lobbying process to effect legislative changes would not be considered part of the program. Likewise, changes to a supplier's implementation were generally taken as constraints, so our expectation is that a limited set of changes would have a unidirectional effect.

In order to create a map of how changes actually affected the system, we need to derive what we will call a *change DSM* (see [19]). Like the structural DSM, column headings show what Clarkson, Simons and Eckert [4] call the 'instigating' area, and rows show the

'affected' area. While individual change requests are recorded for the system as described above, the set of those changes which were associated with other requested changes must be defined, and then described based on the areas affected by each change request.

In Chapter 3 we will see that if we have a change request for Area 5 which describes a change request for Area 1 as its parent and a change request for Area 8 as its sibling, then we can determine that the Area 1 change propagated to Area 5, and most likely to Area 8 as well. However, it is possible that when we look at the dates and detailed information for the change request for Area 8 we will find that it had a completion of -1, and was closed before the change to Area 5 was opened, in which case the better conclusion is that a change was made to Area 5 instead of Area 8 to finish resolving the problem which caused the change request in Area 1.

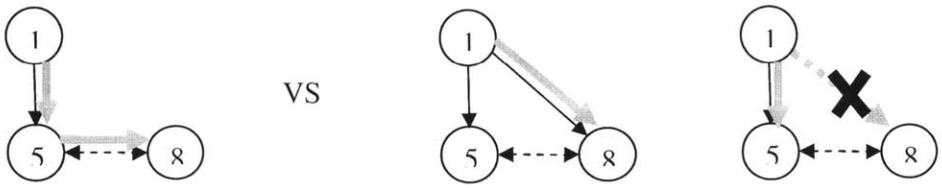


Figure 12: Propagated Change versus Substituted Change

In order to consolidate the information as needed to describe the propagation of changes between different areas of the program in contrast to the expected structural mappings, we imported the simplified change report entries into a Matlab tool. A listing of individual edges between nodes (data records) was then created, along with an area by area summary. Each edge indicates either a parent-child or sibling relationship between two change requests. In addition, the change magnitude of the destination node was correlated with each edge.

This combined data allows aggregate frequency and impact values to be calculated for changes propagating within and between each of the areas of the program.

2.9 Creating the Change DSM

Initially, the frequencies of propagation were computed only for edges contained entirely within the first 25,000 entries, yielding a matrix from which the following is excerpted:

Table 5: Propagation Frequency Excerpt

Area	1	2	3	4	5	6	7	8	9	10
1	0.4843	0.0011	0.0136	0.0057	0.0125	0.0023	0.0079	0.0000	0.0028	0.0040
2	0.0061	0.0000	0.0000	0.0030	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
3	0.0173	0.0000	0.1053	0.0050	0.0012	0.0000	0.0000	0.0000	0.0012	0.0037
4	0.0224	0.0000	0.0112	0.0449	0.0000	0.0000	0.0075	0.0000	0.0000	0.0150
5	0.0137	0.0000	0.0000	0.0000	0.1262	0.0000	0.0000	0.0000	0.0000	0.0027
6	0.0417	0.0000	0.0000	0.0000	0.0000	0.0833	0.0000	0.0000	0.0000	0.0000
7	0.0160	0.0000	0.0000	0.0053	0.0000	0.0000	0.0507	0.0053	0.0000	0.0080
8	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0423	0.0000	0.0000	0.0000
9	0.0276	0.0000	0.0138	0.0000	0.0000	0.0000	0.0000	0.0000	0.0138	0.0000
10	0.0138	0.0000	0.0046	0.0138	0.0046	0.0000	0.0138	0.0000	0.0000	0.1244

Note that, depending upon the nature of the area, there might be no internal propagation (knock on changes), or they might be very common. Area 1 contains the requirements documentation- both at the program level and at the software and hardware item level. Nearly half of the time that requirements type changes were made within this subset they were associated with another requirements change (also within Area 1). In contrast, area 2 contains other programmatic documentation, including software detailed designs. Due perhaps to these typically being changed only during the detailed design phase, the instances of follow-on changes are rare (thus the value of 0.000 where row 2 and column 2 intersect). However, upon further investigation of the overall frequencies for area 2, they show that the vast majority of changes made in that area were insufficiently documented in the change management system, so more accurate understanding of the cause is difficult to achieve.

The overall pattern yielded (with any value greater than zero being indicated as a link- a shaded cell) from the connections within the first 25,000 entries is the *initial change DSM*:

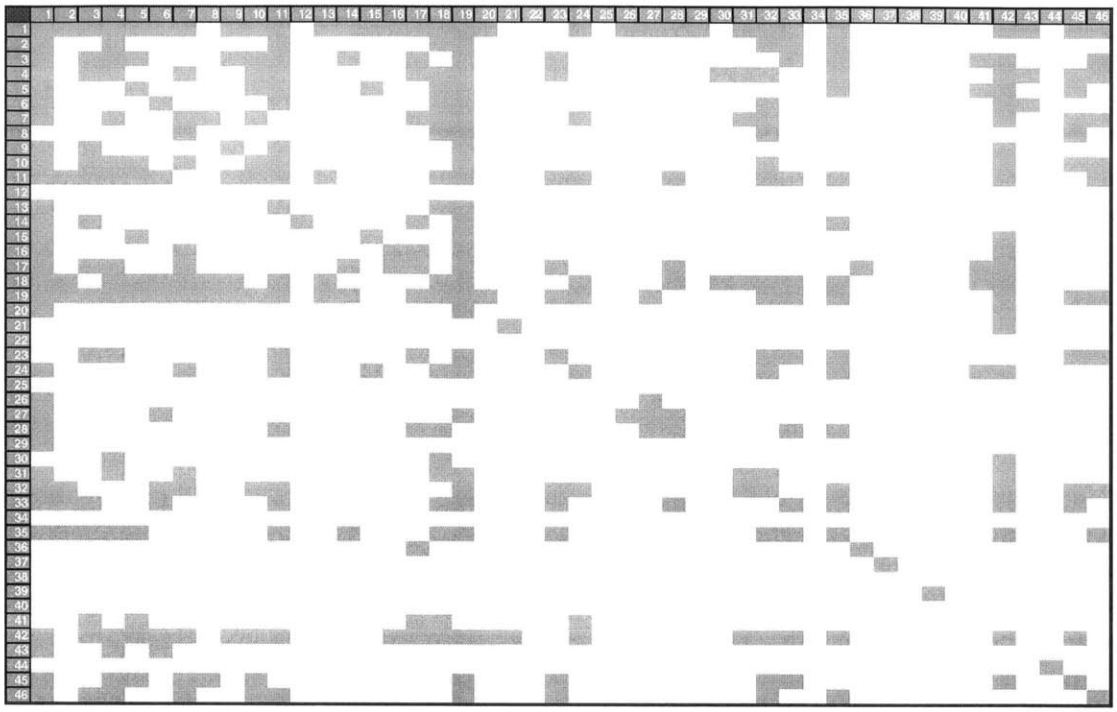


Figure 13: Initial Change DSM

Comparing the initial change DSM (and the full data set change DSM) to the structural DSM shows some fascinating patterns. These will be investigated and described in the following chapter.

2.10 Change Prediction Model Representation

In addition to the previously discussed depictions, the data was used to form an ex post change model in the CPM tool created by the Cambridge EDC (see [2]). In order to do this, the likelihood of propagation occurring from one area to another was computed as a simple proportion by counting the number of instances where changes were linked from area m to area n, then dividing by the total number of changes in area m. One interesting facet highlighted by this method is that in some cases one change causes more than one change in a particular area (and, in fact, it is most likely that one change will cause more than one change in the same area, and in one instance changes in an area *on average* caused more than one change in that same area). In order to include these cases in the tool, likelihood was set to 1, and the impact values were correspondingly increased such that the overall risk value is consistent. This simplification would be a good one to reconsider for future work.

Traditional methods of change management might not depict this, however the granularity available from the data reporting on the program (where a change request was written for cohesive sets of changes within one area, so that additional alterations not in the spirit of the original change request would be recorded separately) highlight the fact that change propagation is not simple to characterize for a system of this size- in particular, the 46 program 'areas' chosen for modeling are more akin to subsystems than

traditional individual components, allowing the possibility of multiple distinct changes being necessary to change a characteristic function of the system.

As a result of the combinations of data which were available for extraction, and the different methods for displaying and interpreting that data (including use of network depiction, DSMs, and the EDC's CPM tool), several interesting patterns emerged for consideration, and are described in the following chapter.

3. Results

Once assembled, patterns began to emerge from the data. One of the first was the low frequency of propagation, overall, from one area to another. For links contained entirely within the first 25,000 entries, actual frequencies for inter-area change relationships only occasionally exceeded 2%, and very rarely exceeded 5%. In the full data set the frequency increased slightly, but still only exceeded 10% propagation in just over 2½ % of the potential cases. This is particularly interesting in relationship to other work focused on mechanical systems, where in some cases it can be considered that in the vast majority of cases either a change will occur or it will not: in many ways leading to much easier modeling and prediction. Our hypothesis is that the architecture of this software dominated system was carefully crafted to be modular from the start, thus leading to less frequent inter-area change propagation compared to a purely mechanical system or non-modular software.

On the other hand, a larger number of areas than might be expected were connected by propagation- just at low rates. The DSM views below are the result of overlaying the structural DSM with 1) the change DSM for the first 25,000 entries and 2) the change DSM for all of the entries. The medium gray squares labeled 'p' are those where a structural connection was known (thus change propagation was predicted) and change propagation actually occurred. The light gray squares labeled 's' are those areas where change propagation (to the 4^{th} decimal place) did not occur, yet a structural connection was present. The dark gray areas labeled 'C' highlight those places where a direct structural connection was not noted, but change propagation *did* occur.

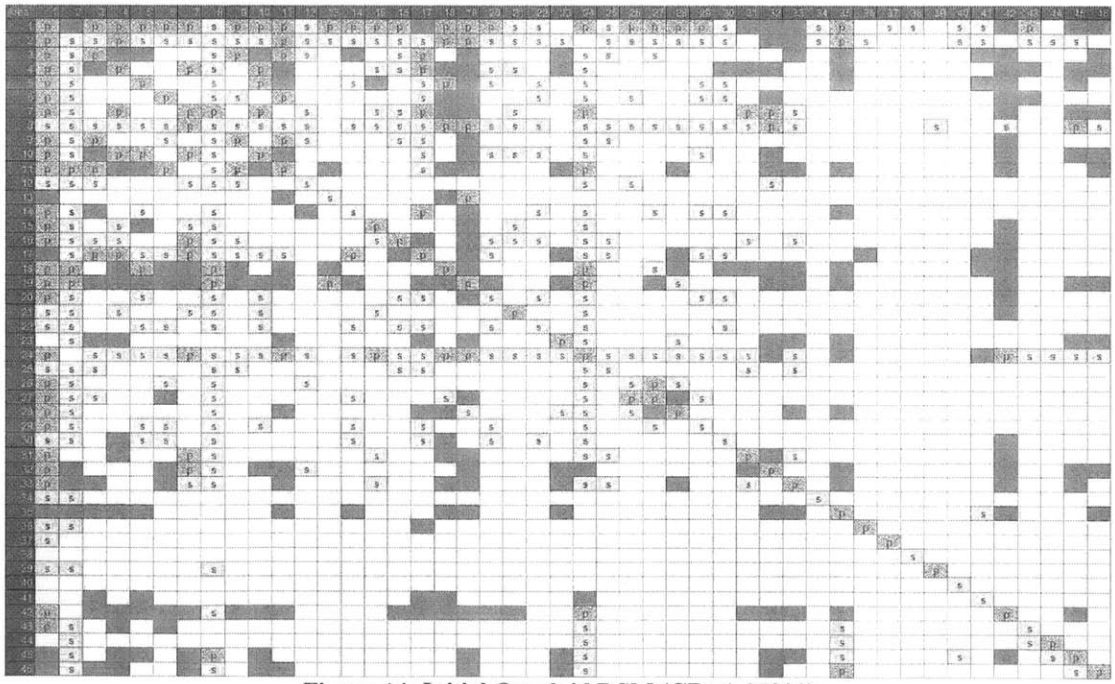


Figure 14: Initial Overlaid DSM (CRs 1-25000)

Interesting to note between the two figures (Figure 14 and Figure 15) is the expansion of the areas of unexpected propagation as the program continued- as changes progress the natural buffers (margins [1]) in the design are decreased, and changes begin to propagate further. Unanticipated problems are discovered during system integration and testing and are resolved, causing more inter-area change propagation. The exception to this is in the case of the components most expected to propagate to many components, there are more instances of structural connection but no propagation connection! (Note Areas 8 and 24.)

Figure 15: Final Overlaid DSM (All CRs)

The lack of propagation in the highly connected areas points to the importance of knowledge. In this data, documented structural connection is available as a proxy for the awareness level of the program team of connections, and is a strong contra-indicator for propagation. The most intuitive explanation is that changes known to be risky in terms of propagation get more attention from the beginning, and therefore end up being resolved sooner. In addition, after a few cycles of high pressure time periods when work is accelerated to meet an interim goal, the interfaces migrate from their strictly designed definitions. In the push for rapid execution engineers occasionally break the rules to get the functionality out of the door a little bit faster. This leads to the classic rework loop described by System Dynamics, where less than perfect quality leads to some quantity of unknown defects, which mean that later work is based on work containing defects, and those defects must be fixed at some point, adding to the total quantity of work to be done in the course of the program [16].

3.1 Patterns: Change Focused

Many other patterns emerged from the data as analysis proceeded. Initially, a search was performed for the largest change component (set of mutually linked changes). A non-exhaustive search revealed a largest component of 2579 linked changes, while the

majority of the components found were comprised of 15 to 60 changes. Given the size of the largest network, it was realized that the goal of analyzing several of the largest components would be beyond the scope (and time available) for this thesis.

Table 6: Five Largest Components

Five Largest Components	Size of Component (nodes)
1	2579
2	424
3	170
4	87
5	64

Given knowledge of the development of the system, the focus shifted to highlighting those change components containing one or more changes resulting directly from customer direction. A plot of this largest component is included below.

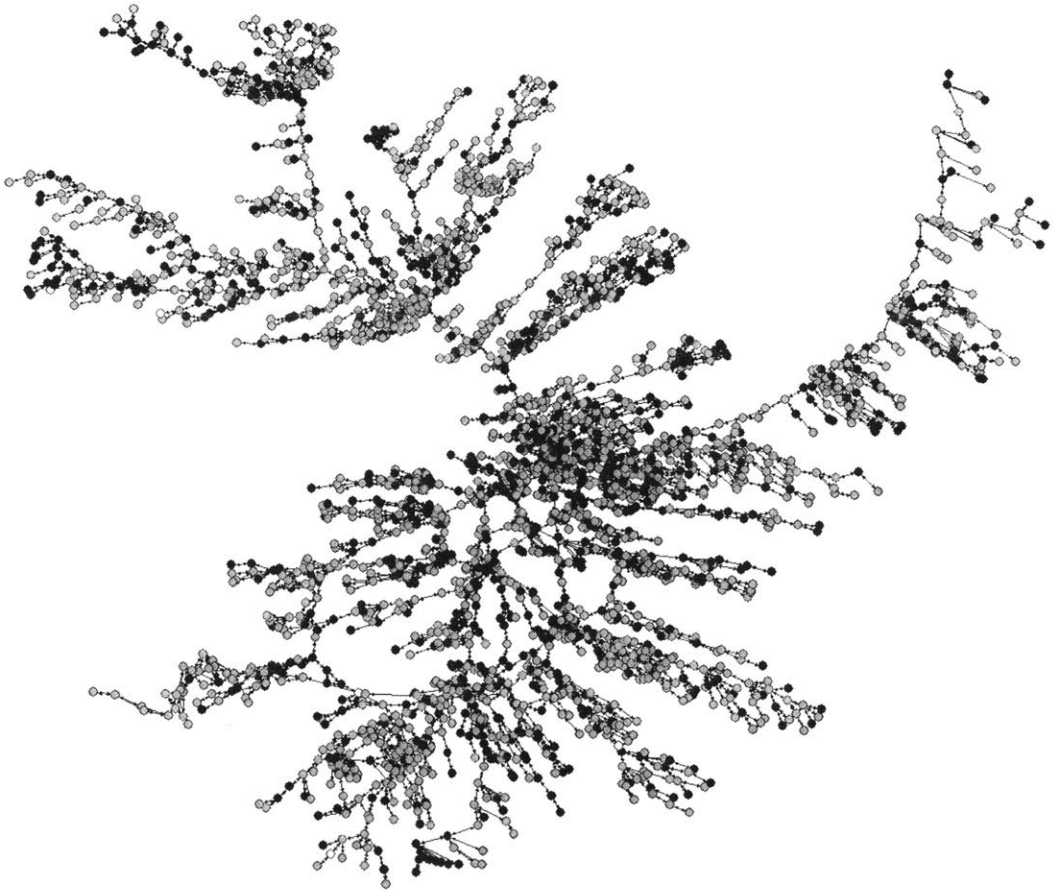


Figure 16: Largest Change Component

Each node in this graph represents a single change request, while the lines linking them depict relationships described in the change requests.

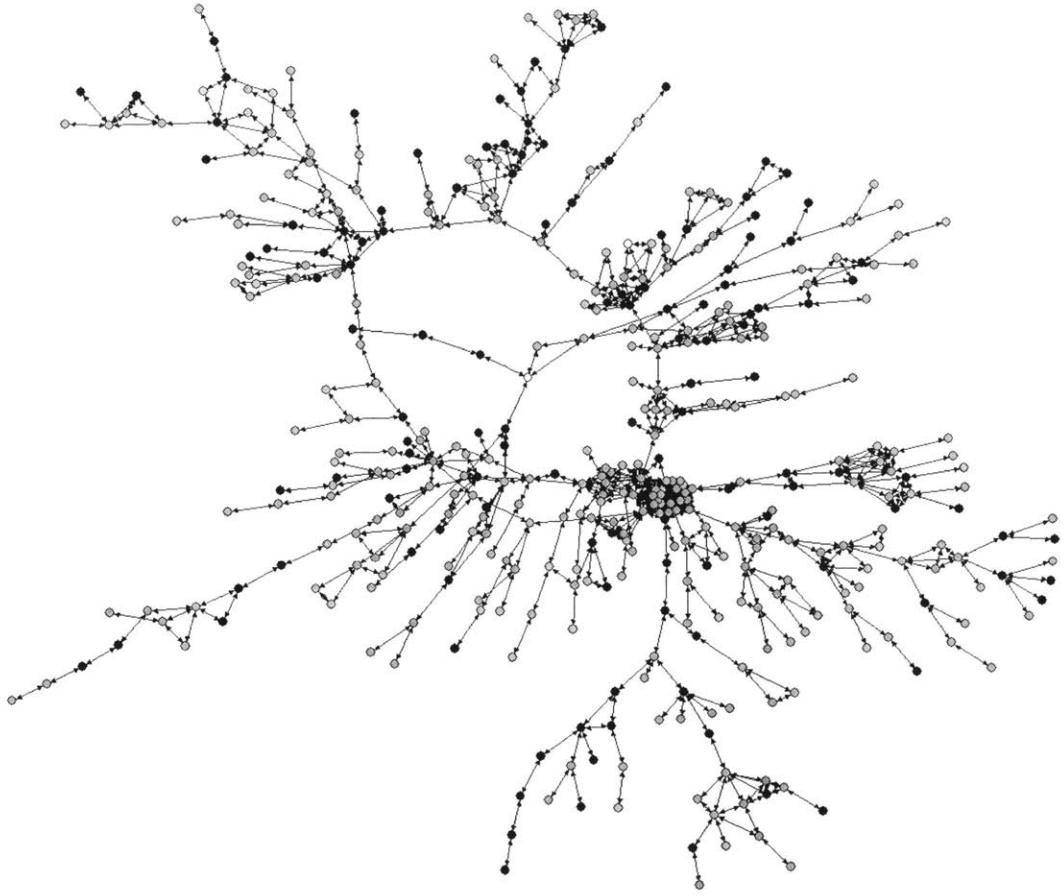


Figure 17: Second Largest Change Component

In Figure 17 most of the nodes are sparsely connected, however a tightly bundled concentration of change requests can be seen just to the right and below center. This configuration implies that these change requests are highly connected, with each change request referring to multiple others and having been worked concurrently.

In Figure 18, the upper left quadrant depicts the elements of that change component which had been written prior to the 20,000th change request in the program. The upper right quadrant shows the component development as of the 25,000th, and the lower left and right quadrants show progression as of the 30,000th and 40,000th change requests respectively. The final component as of data capture is shown in Figure 19, with only the open (white with a black outline) change request having been added after the last snapshot at top center. This sequence shows us the growth of individual change components, which gradually become connected and then expand through the patterns described later in this chapter (see Figures 23 and 24), over the course of the project.

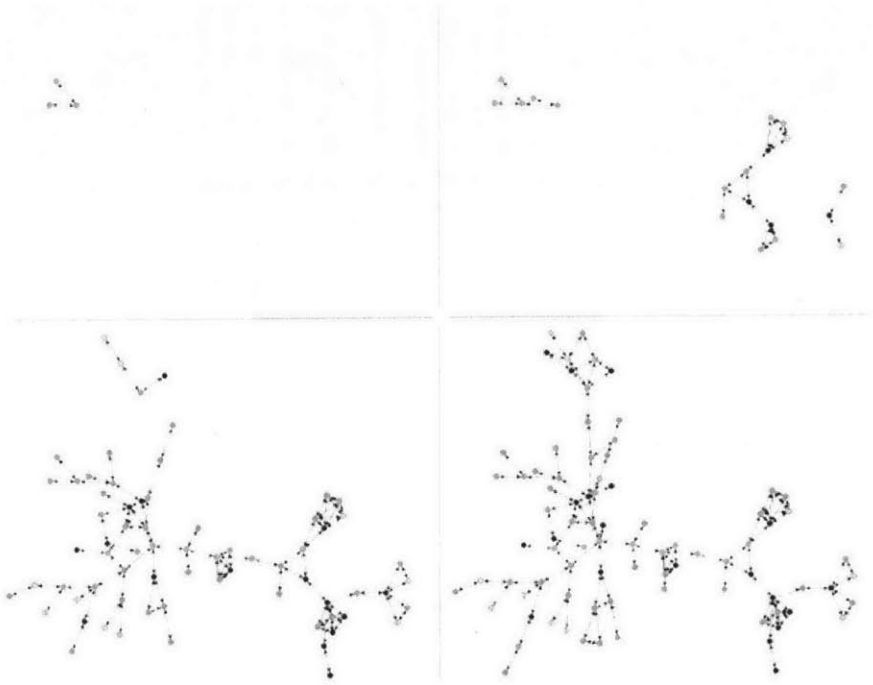


Figure 18: Time Lapse Progression of Fourth Largest Change Component Development

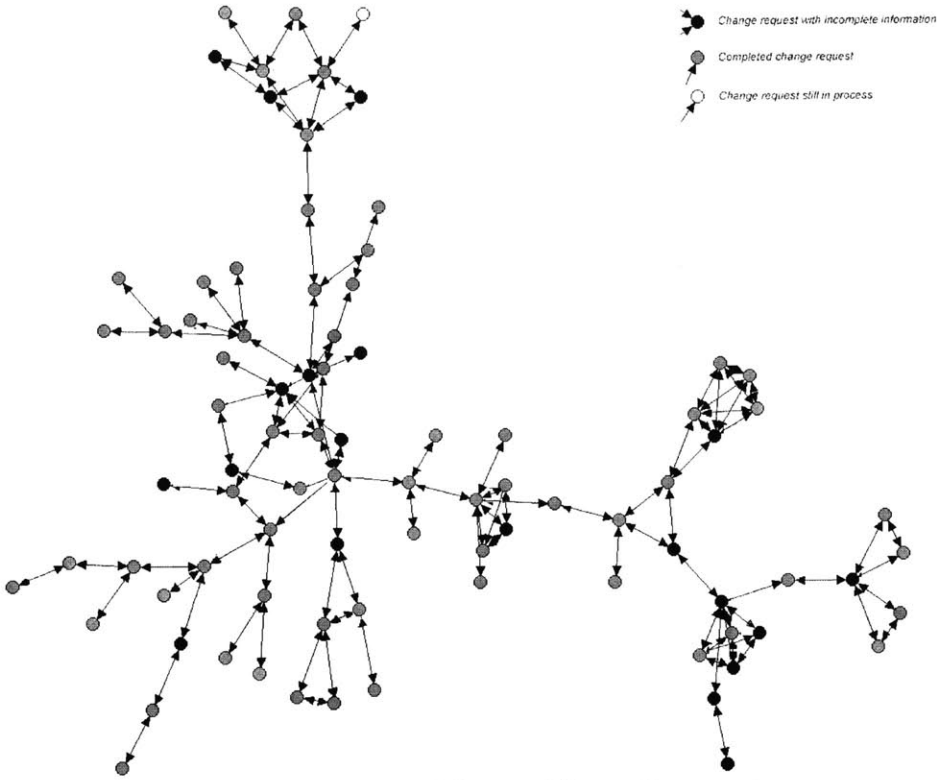


Figure 19: Final Fourth Largest Change Component

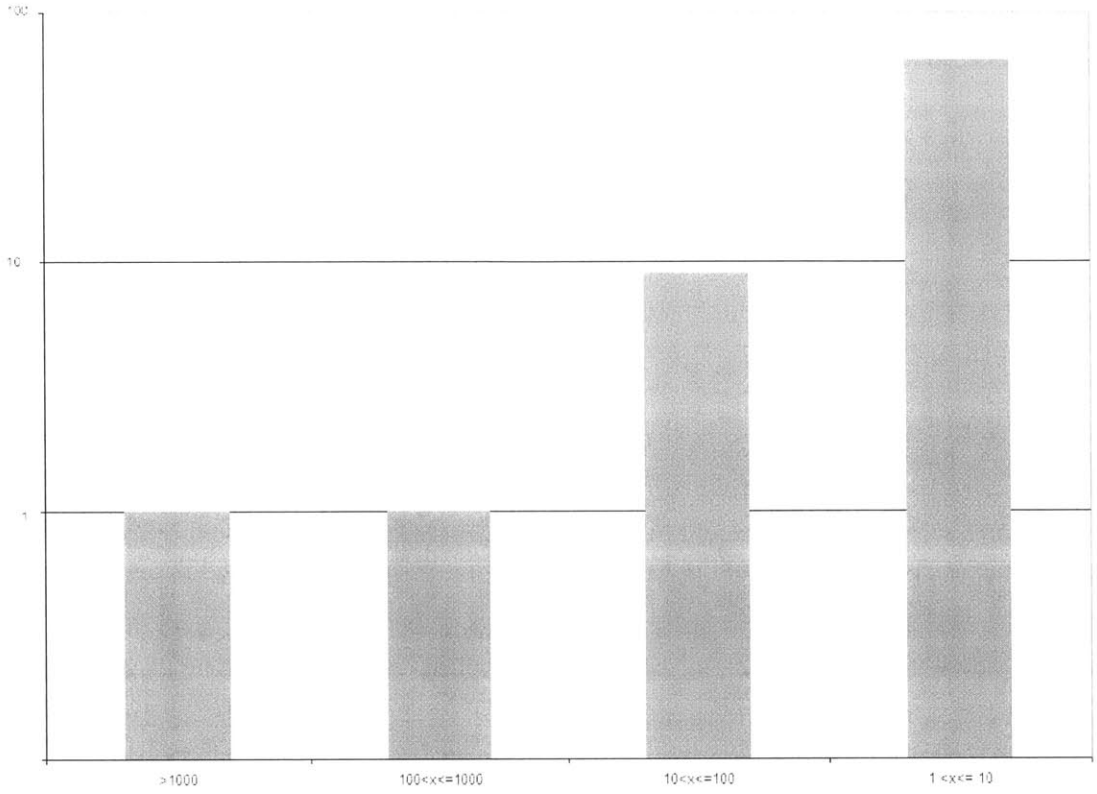


Figure 20: Histogram of Change Network Distribution (Number versus Size)

The notable spread of the largest component led to several questions. First among them, what sort of distribution is present in terms of the number of changes directly connected to any one change?

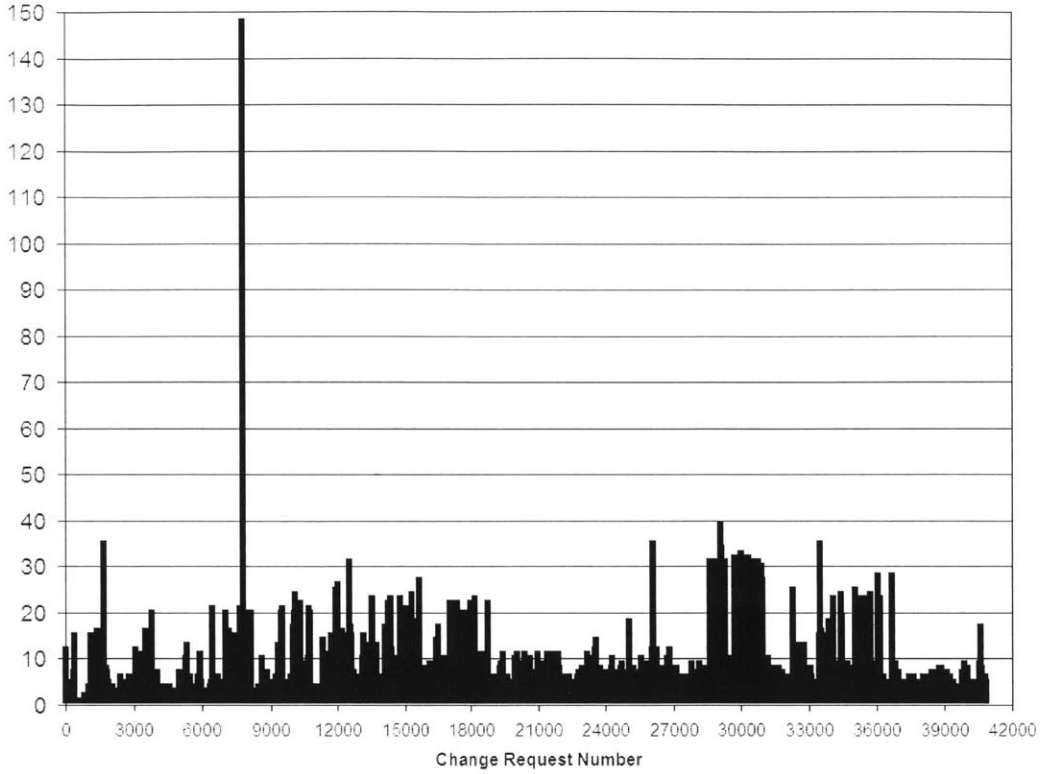


Figure 21: Number of Associated Change Requests

Figure 21 above displays a number of interesting characteristics: first, the vast majority of changes have 10 or fewer direct (first order) connections, however there are a significant number, occurring in the in the first half of the program with a fairly even distribution, which have 10 to 25 connections. In contrast, there are significant periods where almost none exceed 10, punctuated by a select number with much higher connectivity- the distribution becomes bimodal.

Secondly, can some of the smaller component graphs be usefully traced? How are they distributed over time? The largest component would be too unwieldy once change identifiers were added, however the following graph is an illustration of the third largest of the change networks (ordered by number if changes), and it should be noted that it contains a change request which was noted as a direct result of a customer request (indicated by change # 30742).

The areas affected by each change request are noted where possible, and the overlaid arrows trace the progress of the changes as they occurred. The light gray node was a change request still in process as of when the data set was collected, striped nodes indicate change requests which were closed as not completed (withdrawn, disapproved, etc), while dark gray nodes indicate completed change requests, and black nodes indicate those where completion information is not available.

Tracing the progress of nodes in ascending order of change request (and thus by opening date) shows us that these components include the convergence of smaller components, with a set of final solutions truncating several sets of earlier changes in some cases (the solutions being designed to satisfy multiple problems in a cohesive manner). Here the majority of the changes were unearthed by internal testing, although a customer reported change which is part of the network shows up later in the process.

Figure 22: Initial Change Paths in Fourth Largest Component

The pattern this appears to show is one of concurrent discovery of related problems, however we need to evaluate further data- the finish dates of the changes- in order to get the whole picture.

Most generally, there are two overall patterns recurring throughout the structure: inside out, or solution divergence, where a single change request is the origin of several; and outside in, or solution convergence, where multiple (initially disjointed) change requests are connected through a common fix.

Figure 23: Overall Patterns

In addition to the overall divergence and convergence, there are three canonical patterns which emerge on the small scale: nominal, knock-on, and superseding.

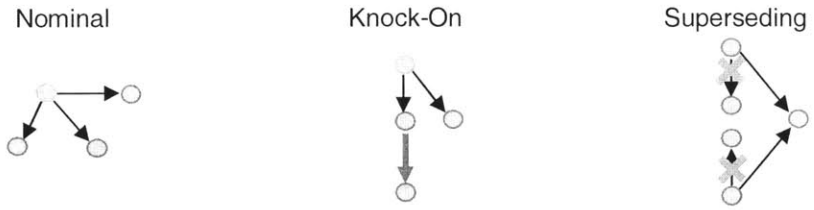


Figure 24: Canonical Patterns

In the nominal pattern a parent change request is submitted, followed by one or more child change requests. These are all incorporated into the project and closed at the same time. In the knock-on change pattern, a parent and child requests are submitted, but at least one of the child requests generates knock-on requests. The superseding pattern occurs when one or more initial approaches are abandoned in favor of other solutions (change requests). We expect that there are other canonical sets which could be described, which would be another avenue for future work.

When pursuing the re-evaluation of the mapping based on closing date [Figure 25], we see that the change requests first opened are not necessarily those first closed. Tracing based on closure date when adjacent change requests have overlaps in the periods during which they were being worked, we see a different pattern. The top segment actually spreads down and accounts for much of the center segment of the overall change component, rather than some combination of smaller components linking up sequentially. Much more of the change component was worked concurrently and with different effects than would be guessed from evaluating the causes and effects based solely on change request opening date.

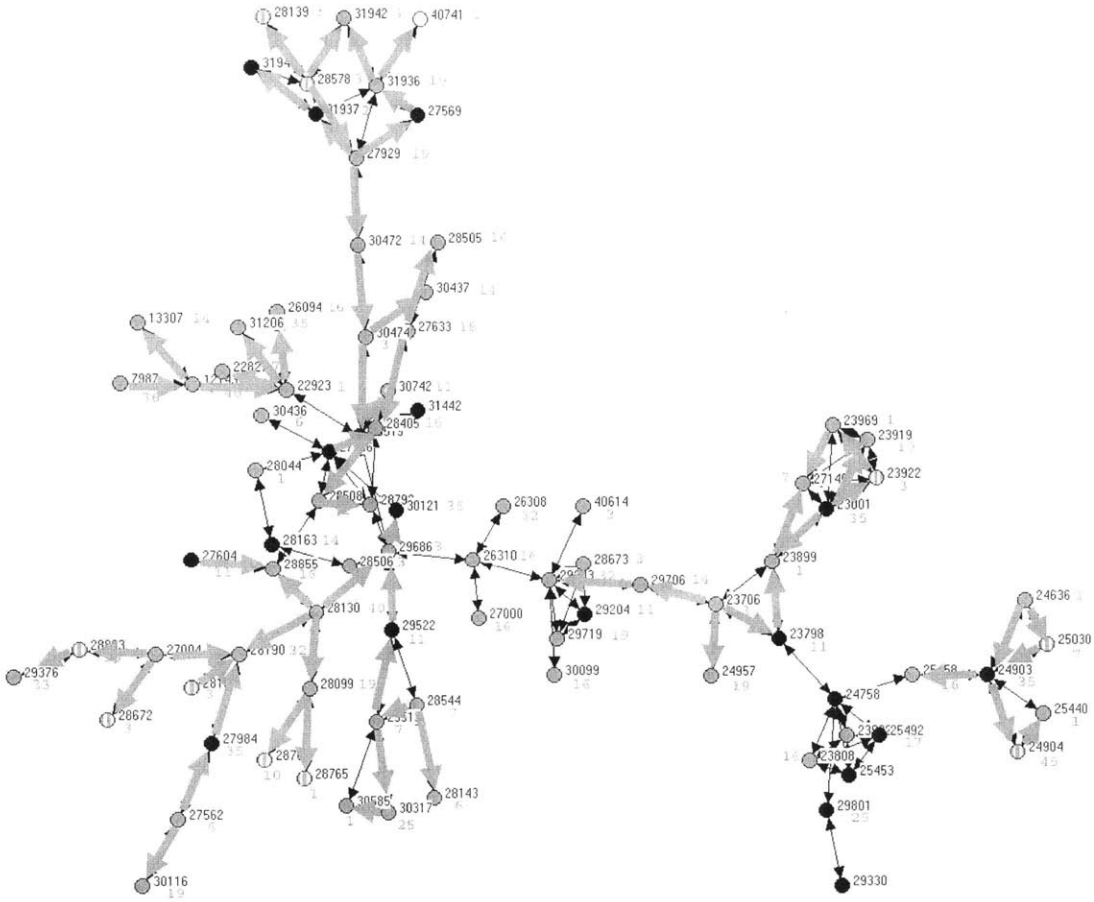


Figure 25: Remapped Change Paths in Fourth Largest Component

One additional interesting aspect to note from this tracing is that in this case we have stumbled upon some quantitative evidence related to observations made by Clarkson, Simons and Eckert regarding some rough rules of thumb in change propagation analysis:

> "The change propagation analysis was performed assuming that changes would not propagate appreciably beyond four steps. Further analysis of the model showed this to be a reasonable assumption."[4]

This finding appears to be largely confirmed as a rule of thumb by the data seen in this thesis. Based upon inspection of the largest change components, the vast majority terminate within four causal steps of an originating initial change. However, as some more extended paths do occur, for a program of such extent and complexity as that analyzed here it would be wise to retain additional detail and re-check such assumptions on a regular basis.

These graphs show fascinating patterns for individual changes, but how do these aggregate?

3.2 Patterns: Area Focused

With the impact percentages derived from the summaries of the edges (where an edge represents a parent-child or sibling relationship between two change requests), combined with the total number of completed changes for each area, we can derive an understanding of which components are multipliers, absorbers, carriers and constants. Some edges are contained within a single area, while others are between nodes, or change requests, in different areas. In addition, parent-child relationships are unidirectional, while sibling relationships are bidirectional (they can be characterized as a changes to I and J both affecting each other). We can define the percentage propagation times the number of completed changes for the originating area as the change value for area I affecting area J, the sum of change values from all 46 areas affecting area I as C_in (changes in) and the sum of change values from area I as C_out (changes out). The difference between these is a reasonable posterior approximation of the Change Propagation Index (CPI) as used by de Weck and Suh in [5].

$$C_{IJ} = \frac{(parentchild_{I,J} + sibling_{IJ})}{totalchangerequests_I}$$

$$C_{OUT}(I) = \sum_{J=1}^{46} (C_{IJ} \cdot totalchangerequests_I)$$

$$C_{IN}(I) = \sum_{J=1}^{46} (C_{JI} \cdot totalchangerequests_J)$$

where:

$$sibling_{Ix} = sibling_{xI}$$

Note that data was maintained in the form of C_{IJ} and total change requests for each area- this was an artifact of the data extraction process, and the data format caused non-integer values to emerge for $C_{OUT/IN}(I)$. More accurate values long term could be determined by maintaining the parent-child and sibling change request counts individually, which would not be difficult to ensure with proper planning.

After these computations, a normalized CPI can be calculated, showing the relative strength of the component on the absorber-multiplier spectrum.

$$C_{NORM}(I) = \frac{C_{OUT}(I) - C_{IN}(I)}{C_{OUT}(I) + C_{IN}(I)}$$

ex.

$$C_{NORM}(7) = \frac{275.0072 - 210.4849}{275.0072 + 210.4849} = 0.133$$

There is latitude in defining what is a carrier- in this case those components having an absolute value of the difference between C_out and C_in less than or equal to 10% of C_in are identified as carriers. There is no hard technical basis for choosing this number, and a certain amount of tuning could change the proportions of the results significantly, as befits a non-ideal continuously distributed data set with the very components changing (sometimes rapidly) over several years of development.

It is interesting to note that components 27 and 41 are potentially perfect absorbers or absolute reflectors of change in this system: no submitted change requests were outgoing, while a number of incoming change requests were written. Further investigation shows that 0 changes were actually completed for either component, despite a good number being proposed- every time a potential change was evaluated, the decision was made to either not make a related change, or to make the change in another area. Therefore, while the simple interpretation of the numbers indicated that these components are absorbers, we can better characterize these two components as absolute reflectors: any attempts at change were reflected onto other parts of the system.

Table 7: Area Change Classification

Area	C_in	C_out	C_norm	Description
1	4661.664	3913.65	-0.087	CARRIER
2	86.9734	7.718	-0.837	ABSORBER
3	252.0267	516.2211	0.344	MULTIPLIER
4	189.879	69.0948	-0.466	ABSORBER
5	114.801	263.8893	0.394	MULTIPLIER
6	81.3091	87.963	0.039	CARRIER
7	210.4849	275.0072	0.133	MULTIPLIER
8	66.0833	23.3244	-0.478	ABSORBER
9	105.6976	43.38	-0.418	ABSORBER
10	147.0403	128.9644	-0.065	CARRIER
11	276.9874	66.2592	-0.614	ABSORBER
12	67.1952	44.3578	-0.205	ABSORBER
13	17.3361	18.5334	0.033	CARRIER
14	145.0987	233.9064	0.234	MULTIPLIER
15	237.5956	75.7904	-0.516	ABSORBER
16	303.9338	1325.2064	0.627	MULTIPLIER
17	93.4245	20.413	-0.641	ABSORBER
18	178.0827	23.4702	-0.767	ABSORBER
19	283.5913	586.1024	0.348	MULTIPLIER
20	89.8246	67.608	-0.141	ABSORBER
21	61.0307	36.207	-0.255	ABSORBER
22	47.103	25.0665	-0.305	ABSORBER
23	146.7873	132.5286	-0.051	CARRIER
24	117.7198	35.0536	-0.541	ABSORBER
25	48.8033	160.6176	0.534	MULTIPLIER
26	5.5755	1.2726	-0.628	ABSORBER
27	27.4746	0	-1.000	ABSORBER
28	62.7338	12.428	-0.669	ABSORBER
29	59.4636	7.7217	-0.770	ABSORBER
30	79.3744	20.7366	-0.586	ABSORBER
31	34.6836	50.8695	0.189	MULTIPLIER
32	186.3593	404.5685	0.369	MULTIPLIER
33	93.2965	20.601	-0.638	ABSORBER
34	0	0	0.000	CONSTANT
35	87.9733	58.31	-0.203	ABSORBER
36	59.5611	58.7112	-0.007	CARRIER
37	1.7142	1.7142	0.000	CONSTANT
38	0	0	0.000	CONSTANT
39	24.7026	15.5554	-0.227	ABSORBER
40	3.899	3.6	-0.040	CARRIER
41	4.3617	0	-1.000	ABSORBER
42	107.148	89.8872	-0.088	CARRIER
43	68.195	52.77	-0.128	ABSORBER
44	76.0286	33.561	-0.388	ABSORBER
45	38.5994	42.5776	0.049	CARRIER
46	99.4317	95.832	-0.018	CARRIER

The distribution of the areas on the absorber-multiplier scale can better be visualized in Figure 26 below, and these classifications are discussed in more detail in the following chapter.

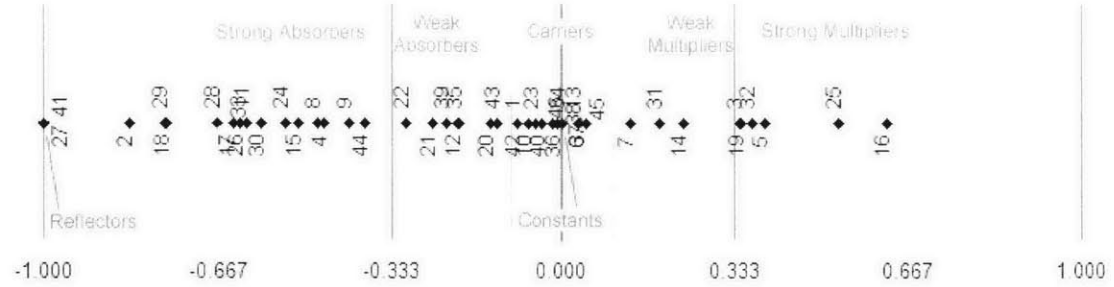


Figure 26: Areas on Normalized Absorber-Multiplier Scale

After translating the area characterizations atop a structural/change DSM through color coding the column and row headings, we achieve Figure 27.

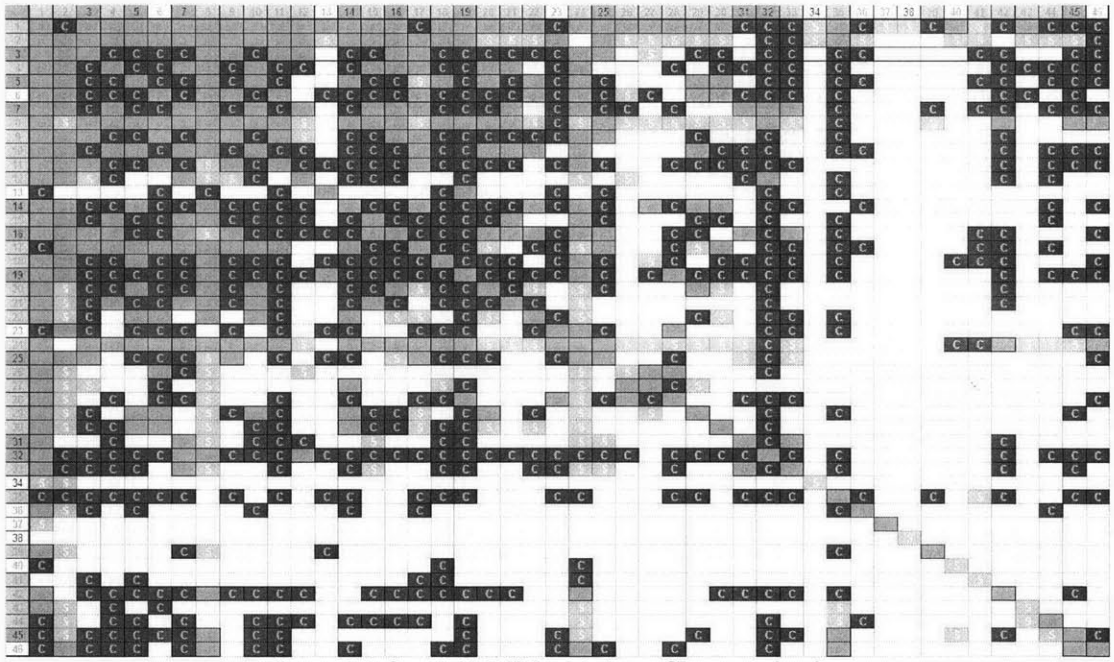


Figure 27: Overlaid DSM with Area Characterizations

Like the composite DSMs above, the letter 'S' (white on light gray) indicates expected structural connection, but no changes. Changes predicted by structural connection are indicated by 'p' (medium gray), and changes not predicted by structural connection are denoted with a 'C' (light gray on dark gray). The information from **Table 7** is encoded in the row and column heading formats using the same color coding as is used in that table.

Sure enough, we see that the overall value for areas 8 and 24, which were denoted in the detailed design material (here shown via the structural DSM) as highly structurally connected saw rather less change propagation than expected, and exceedingly little

unpredicted change. Meanwhile, a glance at area 19 shows that not only did we not predict connection with other areas, but those connections did exist and had significant impact on the number of changes required to be processed during the system's development.

Some interesting questions arise concerning areas 4 and 35, which both come out of the numerical computations as absorbers, yet have a significant number of 'C's in their columns. Investigation into the actual propagation frequencies shows that, in the case of 35, the frequency is 6.35% for propagating to area 1 (a propagation predicted by the DSM), but often less than 1% or even 0.5% for propagating to the unexpected areas. Area 4 sees 7.31% propagation to area 1, but often less than 0.3% to the unexpected areas. These low frequency, but still present, propagation patterns are probably the result of the software/data heavy design of this system. In many cases it is possible (if not desirable) to directly logically connect two components which were significantly removed from each other in the design- it would be very difficult to do the equivalent in a mechanical system (the equivalent of running a drive shaft from one corner of the room to another, detouring around all sorts of equipment already in place).

Looking at the macro level with this amount of data allows for some qualitatively different types of observations with regard to change propagation than have been made before. Returning to the question of how change request generation and propagation played out over time, are there any important aspects to note?

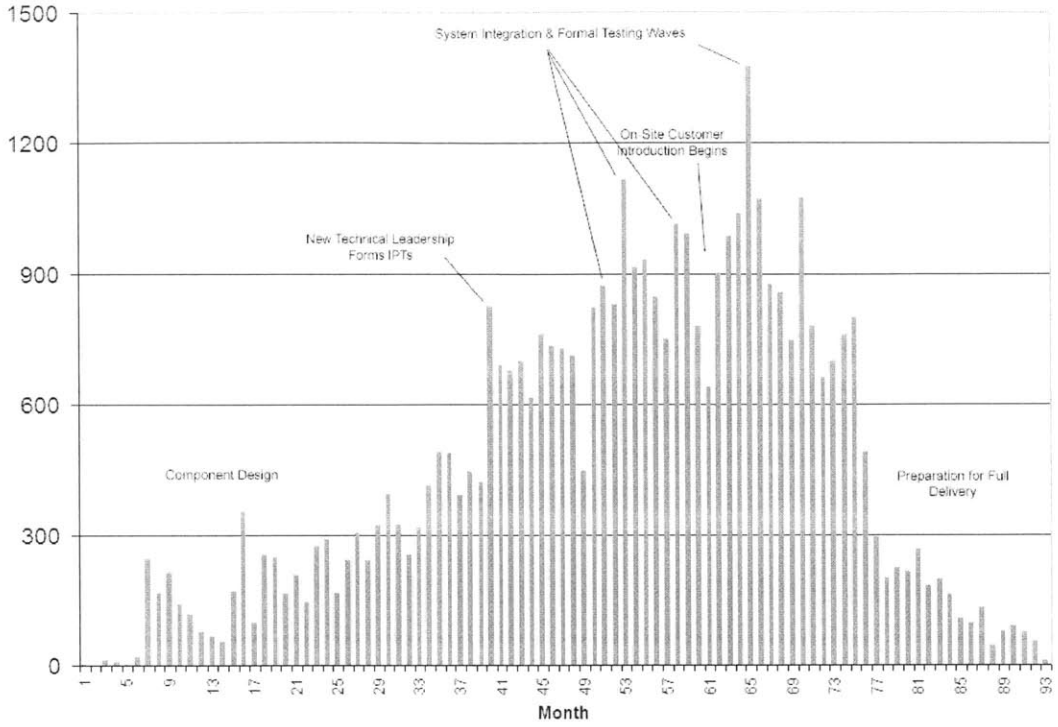


Figure 28: Change Requests by Month with Program Segments

An important aspect to reconsider is the rate at which change requests were being generated. The graph above is annotated with some of the key program management time periods and happenings. Features of particular interest include not only the overall curve, which is very similar in shape to that expected of staffing levels for a large project, but the ripples atop the curve. Coinciding with the creation of new inter-functional teams (Integrated Product Teams [IPTs] to use the common terminology), we see a marked change in the rate of change request generation.

The most salient feature of an IPT is the incorporation of a wide range of disciplines into regular evaluation of a project. Here, around month 40 the project shifted from just those who were working in their own functional areas of development being able to see and question their components to simultaneous evaluation by those concerned with test and system level integration. It is unsurprising that the number of change requests would climb significantly, with the introduction of fresh eyes and different assumptions to the mix.

The overall pattern seen here of accelerated discovery of problems (and thus rework) corresponds with a critical project management recommendation out of System Dynamics [16]- while adding additional personnel to development may just make the project later and further behind budget, a useful way to add personnel is to focus them on discovering rework earlier and more thoroughly.

This begs the question: would the same overall pattern of a reverse or late ripple have occurred without the move to IPTs and a later phased testing regime? Is this a macro level change pattern to be expected when dealing with large aggregated numbers of changes?

It stands to reason that a large and long running program might progress through cyclical phases during which the focus shifts from implementation to testing to fixing and on back. The effects in terms of rate of change request generation would be amplified atop a curve delineated by overall project staffing (more people looking for bugs will probably find a greater number). Some phases will correlate with the tendency to terminate several sets of changes with a single fix- one 'quick fix' to get rid of several problems. While these will lower the total number of changes being written in the short term (and will often correlate with programmatic pressures causing fewer changes in general to be written), it can be expected that such one-solution-fits-all fixes will correlate with additional required changes later on, once deep investigation again gains momentum as the phase shifts back to testing. Potential future work would be to recreate this pattern using a system dynamic model.

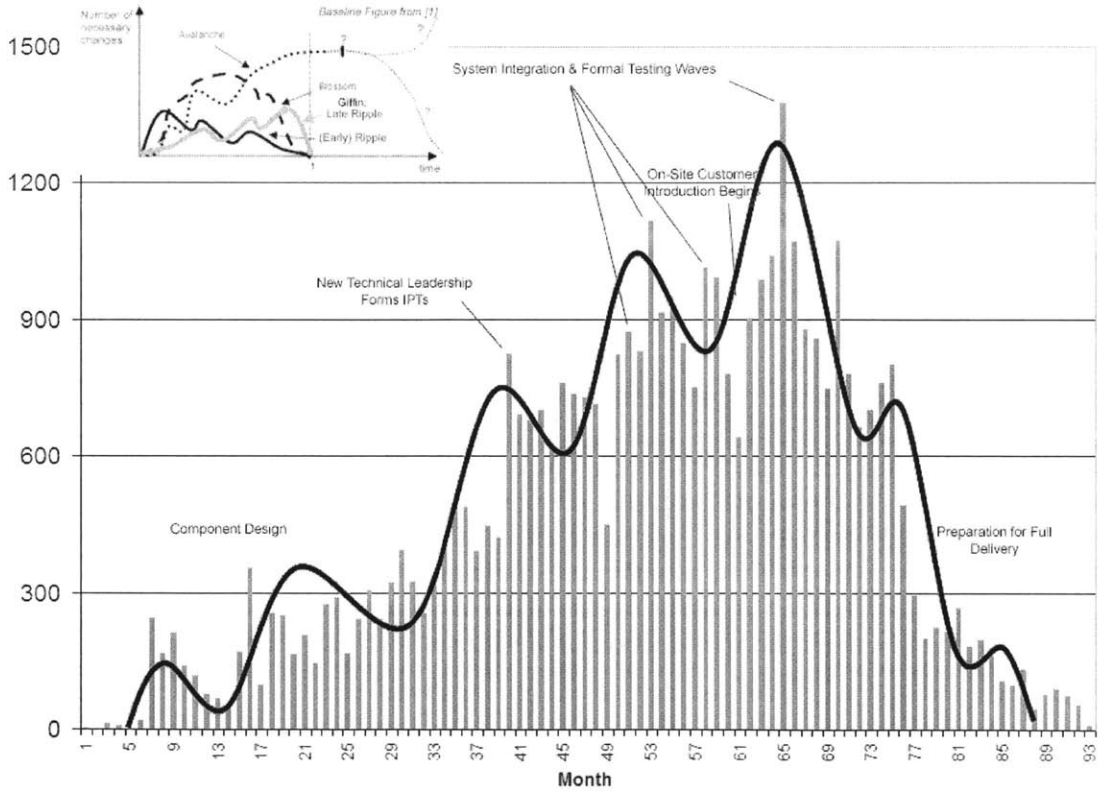


Figure 29: Late Ripple of Change Request Generation

Based upon this data, personal experience, and many discussions with other systems engineers regarding experiences in a large number of programs in a diverse set of industries, the author posits that observed ripple patterns on this scale would correlate with the periodicity of 'political' attention focused on the project, whether from a higher level in the development organization or from customers. Major milestones can act as drivers of change activity in two ways. First, in anticipation of reviews open change requests are closed out in order to present a favorable status at the milestone. Second, a milestone can serve to uncover rework, resulting in new change requests.

The phase correlation of peaks and valleys would likely vary depending on the goals of those who originate the political attention (customers focused primarily on quality might induce large peaks during engagement due to focus on participation in testing, while other customers who threaten program termination might inspire attempts to minimize their concerns). This offers yet another avenue of potential future inquiry.

A question of more immediate interest (and possibility of being addressed) is the following: do individual subsystems [areas] exhibit the same late ripple effect?

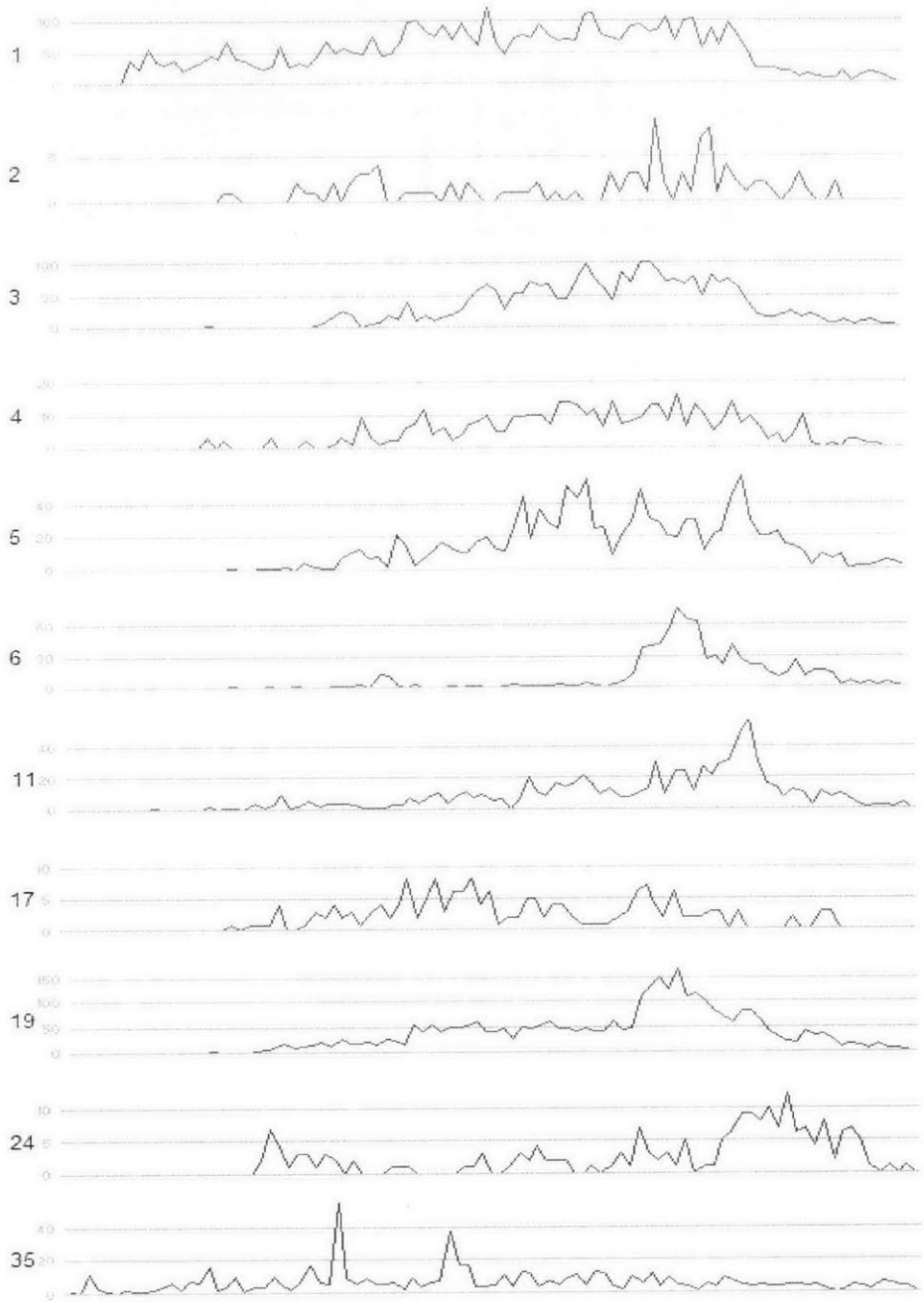


Figure 30: Selection of Area Change Request Sparklines

A quick glance at the levels of change activity throughout the course of the program (for the first six areas, as well as others chosen for their potentially interesting behavior) shows quite a bit of variation. Note that the sparklines above are scaled to emphasize relative patterns, rather than placing them on the same scale which, would emphasize relative amounts of change and make it more difficult to compare overall patterns. Areas 3 and 19, which reach a peak of more than 100 changes generated per month and have high rates of unexpected propagation (are multipliers). It is worth noting that overall there

seems to be no direct correlation between the classification of an area and the change activity pattern, however those very high activity multipliers appear to exhibit some similar traits to the inverted ripple of the overall curve. On the other hand, other areas (most with much lower levels of change request generation) are far more consistent or generally less formed over time. This points towards the late ripple being an aggregate effect, however it should be tested against other data sets.

3.3 Patterns: Staff Focused

If the areas have distinct patterns, what do we see if we look at the people working on the project? A mapping of the same change network as shown in detail before, but adding either the submitter or first assignee of each change requests is below.

Figure 31: Individuals Generating Fourth Largest Component Change Requests

The most outstanding characteristic of this mapping is how few names are repeated- the vast majority of the change requests here were submitted and worked on by different people! For the 87 change requests, 57 individuals could be identified as having been initial contributors. The largest number of change requests in this component written by a single individual is three, while 37 people wrote only one. Of the eight people who wrote three CRs each, only eng034 wrote them all against the same area- the requirements. Overall, though, this tells us that communication is critical- and a shifting cast of people

working on one logical set of changes may be a negative indicator for being able to control and limit the extent of those changes.

This leads to the classic manager's question: if different people are working on these related changes, how much does it matter who they are? While it is not within the scope of this thesis to investigate this in great detail, we can look at a couple of interesting points.

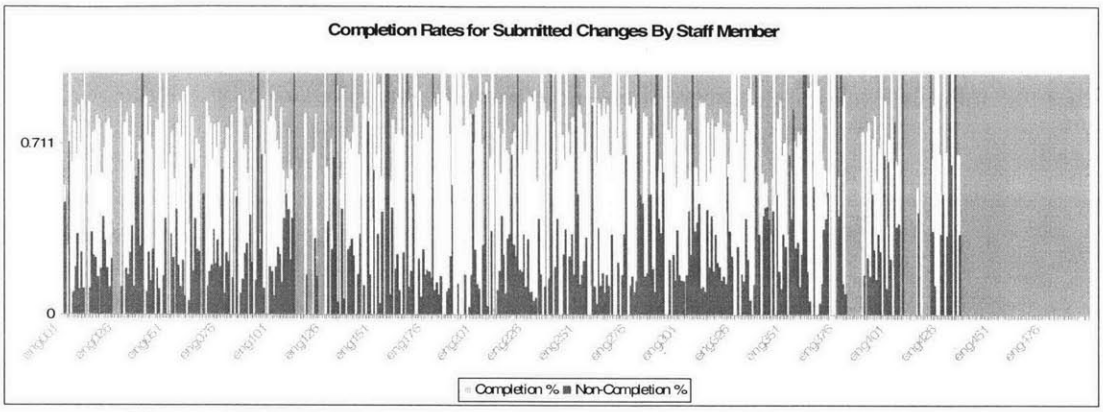


Figure 32: Completion Rates of Change Requests Submitted by Staff Member

In Figure 32 the average completion rate over the whole data set for change requests submitted by a given engineer is 71.1%. A good portion of those engineers involved in large numbers of requests over a significant period of time average at least 80% completion, while other staff members' change requests are never completed. This points to a stratification of staff, and would be useful to map onto the change networks as well, however that remains as an area for future work. See Appendix B for more data on staff involvement in changes over time.

Many pieces of the puzzle have emerged, but how can these be pulled together? What can we learn? These are discussed in the following chapter.

4. Application

As we have seen throughout this paper, changes propagate, and in ways unanticipated by even those individuals most knowledgeable about the system. A comparison of the structural DSM to the change DSM highlights the limitations of our ability to control and predict changes. This program employed multiple intelligent and experienced individuals who spent significant portions of their jobs over many years monitoring these change requests, and seeking to minimize their impact and propagation. Yet, there were *still* many change components with more than 15 elements, with one identified as having more than *2500* elements.

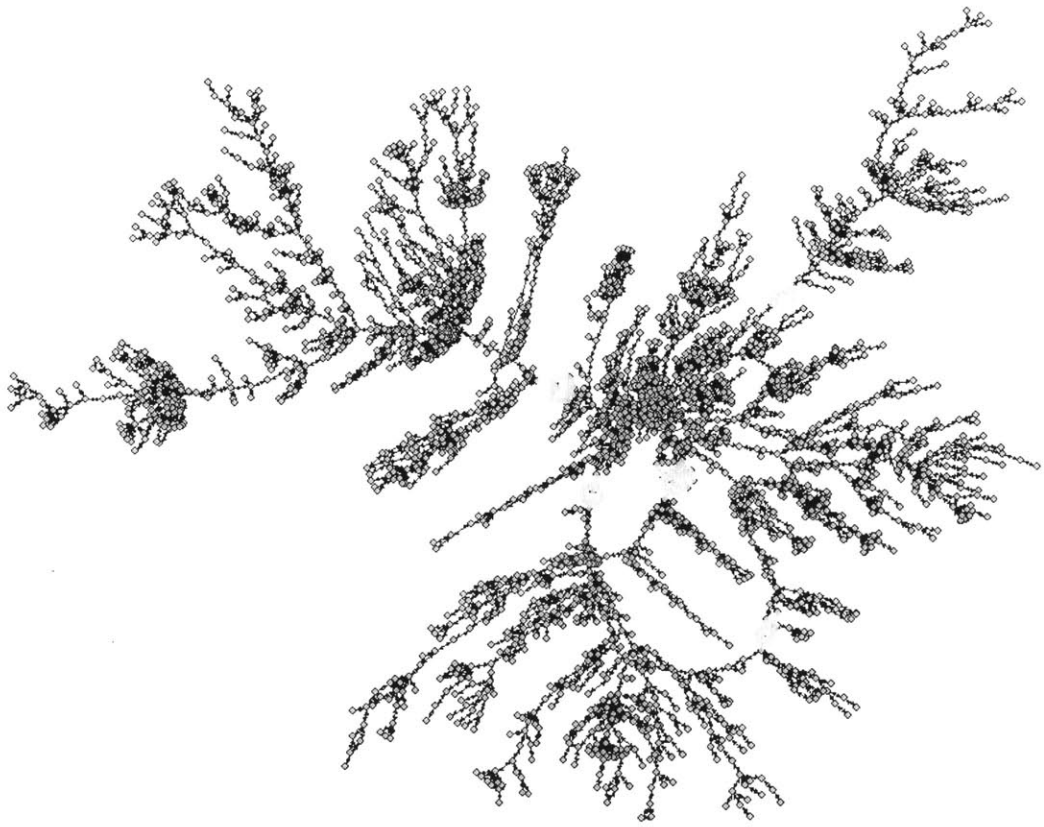


Figure 33: Potential Segmentation of Largest Change Component

In Figure 33 we see the final connected set of change requests which comprise the largest change component. While this seems to imply that a single change could spark hundreds of other changes, it must be understood that this pattern is not a cascade from a single point- rather, many different blooms of change interacted, in some cases being terminated by a single change designed to correct several different issues, in other cases causing additional changes through incorrect implementation or changes in perceived customer needs.

There are several things we can learn from the graph above and the previous chapters:

- During the spread of changes, there are many 'choke points', where a more careful consideration of a single change (or a little additional buffer space for the change) might have prevented a very large number of changes from occurring. Here, terminating the changes at the 5 light gray-circled locations could possibly have cut off the majority of the component, resulting in less complex interplay of change requests.
- While the probabilities of propagation from one area to another are typically quite low in this system, it appears likely that given some number of propagations having already occurred, the probability of the next propagation increases, although in the vast majority of cases there will be four or fewer steps. If this is indeed the case, then explicitly keeping track of propagation may be a strong management tool, and it may be possible to derive a heuristic for what sort of focus is required on a particular change given that it is part of a chain of 'n' length (i.e., if the chain is already greater than four in length, it's time to focus additional effort on understanding the problems at hand).
- Even if components are highly connected, when viewed from the macro level, it is not a given that changes will always propagate along those paths- there are many different technical and non-technical variables at play.
- We should never take it for granted that a change will only affect one, or perhaps a handful of components at the most. While it is highly unlikely that a bounded change in functionality will cascade in such a spectacular way as some segments in Figure 33, it becomes more and more tempting to add a little functionality, or 'simplify' something when making the alterations to satisfy an initial engineering change, leading to higher coupling and unexpected propagation.
- Engineering changes will have knock on effects- the better the team is able to accurately assess these effects in a collaborative manner, the less likely it is that change will propagate.

Another critical lesson from the three largest change components is that there is a very strong tendency to tie different simultaneous changes (that were potentially initiated as separate disjoint changes) in to each other. What start off as unconnected chains of change requests eventually meet up, being terminated (or extended) by a single change request. While useful for speeding up reaction to individual changes, this practice introduces additional complexity to solutions, and probably contributes to rework and further change propagation. This would be another area of interest for follow on work.

4.1 Change Component Analysis: Architectural & Managerial Implications

The change component analysis offers the potential for lessons applicable to architecture and programmatic decision making. In the analysis, six areas stood out as strong multipliers (those chosen here have C_out more than two times C_in). These particular areas are optimal targets for attention throughout the project, in order to attempt to limit change propagation.

Table 8: Strong Multipliers

Area	C_in	C_out	C_norm	Description
1	4661.664	3913.65	-0.087	CARRIER
2	86.9734	7.718	-0.837	ABSORBER
3	252.0267	516.2211	0.344	MULTIPLIER
4	189.879	69.0948	-0.466	ABSORBER
5	114.801	263.8893	0.394	MULTIPLIER
6	81.3091	87.963	0.039	CARRIER
7	210.4849	275.0072	0.133	MULTIPLIER
8	66.0833	23.3244	-0.478	ABSORBER
9	105.6976	43.38	-0.418	ABSORBER
10	147.0403	128.9644	-0.065	CARRIER
11	276.9874	66.2592	-0.614	ABSORBER
12	67.1952	44.3578	-0.205	ABSORBER
13	17.3361	18.5334	0.033	CARRIER
14	145.0987	233.9064	0.234	MULTIPLIER
15	237.6956	75.7904	-0.516	ABSORBER
16	303.9338	1325.2064	0.627	MULTIPLIER
17	93.4245	20.413	-0.641	ABSORBER
18	178.0827	23.4702	-0.767	ABSORBER
19	283.5913	586.1024	0.348	MULTIPLIER
20	89.8246	67.608	-0.141	ABSORBER
21	61.0307	36.207	-0.255	ABSORBER
22	47.103	25.0665	-0.305	ABSORBER
23	146.7873	132.5286	-0.051	CARRIER
24	117.7198	35.0536	-0.541	ABSORBER
25	48.8033	160.6176	0.534	MULTIPLIER
26	5.5755	1.2726	-0.628	ABSORBER
27	27.4746	0	-1.000	ABSORBER
28	62.7338	12.428	-0.669	ABSORBER
29	59.4636	7.7217	-0.770	ABSORBER
30	79.3744	20.7366	-0.586	ABSORBER
31	34.6836	50.8695	0.189	MULTIPLIER
32	186.3593	404.5685	0.369	MULTIPLIER
33	93.2965	20.601	-0.638	ABSORBER
34	0	0	0.000	CONSTANT
35	87.9733	58.31	-0.203	ABSORBER
36	59.5611	58.7112	-0.007	CARRIER
37	1.7142	1.7142	0.000	CONSTANT
38	0	0	0.000	CONSTANT
39	24.7026	15.5554	-0.227	ABSORBER
40	3.899	3.6	-0.040	CARRIER
41	4.3617	0	-1.000	ABSORBER
42	107.148	89.8872	-0.088	CARRIER
43	68.195	52.77	-0.128	ABSORBER
44	76.0286	33.561	-0.388	ABSORBER
45	38.5994	42.5776	0.049	CARRIER
46	99.4317	95.832	-0.018	CARRIER

Graphical User Interface

Core Data Processing Logic

Hardware Performance Evaluation

Common Software Services

Hardware Functional Evaluation

System Evaluation Tools

If we take the core data processing logic (Area 5) to be a given (as any desired changes to the aggregate technical system behavior from a customer perspective should affect and be affected by this area), and focus on the others, a commonality emerges. Each area's main function is to be an interface: between human and software, software and hardware, or between many significantly different segments of software

While we know from the systems perspective that interfaces are crucial, the 'hot spots' highlighted here are both a reminder and clarification. Not only do we have to carefully attend to interface definitions between subsystems or components, we need to attend to the whole of the components which sit as the main conduits for the higher level interfaces. These are the places to include change buffers, hold more reviews, and focus IPTs.

One way of capturing the essence of the problem would be to think of a system not as a series of 'black boxes' with traditionally defined and publicized interfaces, but as black box components and interfaces with shock absorber gray boxes where areas with significant functional difference meet.

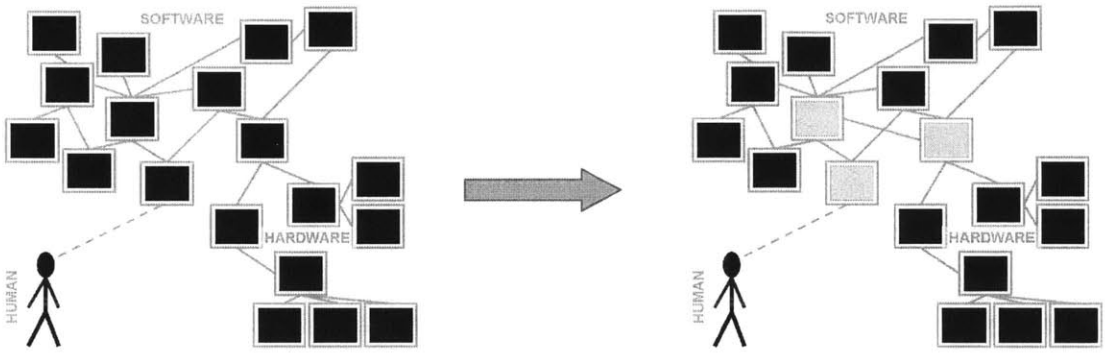


Figure 34: All Black Boxes Versus Some Gray Boxes

The black boxes (components which only interface with a closely functionally related set of other components) can be largely developed and evaluated by the associated functional groups, but gray boxes should be developed and maintained as a collaborative endeavor, with system/architectural level oversight.

Note that the comment above regarding IPTs (or similar cross-function collaboration) does not imply that they should only be used in the gray box interfaces- one thing clear from this project's data is that introducing IPTs around month 40 was associated with significantly accelerated exposure of defects. Systems focused and test focused personnel joined the functional engineers, and the additional perspectives led to reconsideration of assumptions and understanding of inconsistencies between the products of different subsystem teams. We know from System Dynamics [16] that this accelerated identification of rework significantly lessens the overall program length, in large part by shoring up the foundation upon which later work is built. However, given a paucity of personnel to dedicate to IPT activities, those individuals should be focused on the components at the major function interface points (gray boxes).

Other methods which could be used to combat lack of full understanding of relationships within a program include the use of DSMs in design documentation, with subsequent dissemination throughout the program. Old style block communication diagrams are familiar and comforting, but make it very hard to see what is *missing*- places where necessary links are missed. Dissemination of information in that format has the potential to make necessary lines of communication clear both for developing engineers and management.

An aspect to consider in concert with system decomposition and awareness building is how to build in change buffers where possible. While the exact nature of a change buffer will vary widely depending on the technical nature of the program, the technical director or an IPT of technical leads for the program should be responsible for creating and managing change buffers explicitly through inclusion of flexibility. As the program progresses, use of that flexibility should be consciously controlled- just as program manager should be responsible for managing and doling out schedule and cost buffers. Active consideration and continual reevaluation are important here, and should be able to greatly reduce unanticipated and costly change propagation.

If continual reevaluation is necessary, does it need to be at a similar level to this, looking at all of the changes, or can it be using an entirely component level model? At the component model, the existence of propagation connections from one area to the next is what is considered- either area 1 has a propagation connection or area 8, or it does not. If area 1 accepts changes from areas 3 and 5, but only gives changes to area 8, then it is an absorber. In this case, when we follow the lead of previous work and evaluate each component as a multiplier, etc, based on the number of components which input changes versus the number to which changes may be transmitted, it allows the following evaluation:

Table 9: Macro Level Characterization Comparison

Area	Type Based on Total Changes	Type Based on Area Connections Only
1	ABSORBER	CONSTANT
2	ABSORBER	MULTIPLIER
3	MULTIPLIER	MULTIPLIER
4	ABSORBER	ABSORBER
5	MULTIPLIER	ABSORBER
6	CARRIER	ABSORBER
7	MULTIPLIER	ABSORBER
8	ABSORBER	ABSORBER
9	ABSORBER	MULTIPLIER
10	ABSORBER	MULTIPLIER
11	ABSORBER	ABSORBER
12	ABSORBER	MULTIPLIER
13	CARRIER	ABSORBER
14	MULTIPLIER	ABSORBER
15	ABSORBER	ABSORBER

- Require full population of impact as well as review of parent/child/sibling information prior to acceptance of final change
- Associate expected change magnitude and cost, both planned and actual
- Include a simple and *usable* data extraction tool to automatically collate the above information. There has been extensive research in the psychology and human factors domains, and it should be used (see [20] and [21]). The goal here is to encourage engineers to use the tool religiously. Likewise, the output must be in a usable format compatible with common software, etc without editing.

In addition to providing clear and accurate information for further investigation, during the course of the program these could be used for periodic evaluation and course correction.

Given this information, the data will be present to discern where greatest potential for additional propagation and rework lies. The results should be useful in staffing decisions, and continual evaluation of timelines and program health. Excessive linked changes may point to a need to revamp the detailed design information and other documentation available to the team. Additionally, after a short initial work period the information gathered to date should be evaluated in depth, with an active consideration of whether linkages beyond those originally predicted are present or whether enough (or too much) information is being collected.

If difficulties persist in a particular area, investigate creating smaller IPTs consisting of representatives from groups on either functional end of the propagation to review potential changes. In such cases it may be worth investigating redesign to eliminate the change links if making them explicit is not desired.

5. Summary

This thesis was a first attempt at mining the wealth of huge data sets that exist in industry (but are typically hidden from public view) in order to better understand the nature of change and change propagation, and to draw applicable solutions for future development programs of similar scale. Tools currently under development at Cambridge University and the Massachusetts Institute of Technology were used and evaluated in the course of the investigation, and other methods were developed to extract and manipulate the data to yield useful information.

System detailed design documentation was used to create a Design Structure Matrix [DSM] representing the intended structure of the program, then the data was analyzed to yield a change DSM, describing the actual realized change structure of the program. These were compared, and conclusions regarding the combined information were drawn, including:

- Change propagation was a consistent occurrence in the development of this large technical system
- The patterns of change propagation exhibited over the course of the program did not correspond to the component connections described during design
- The detailed patterns of change propagation did not correspond to a component level only propagation characterization
- The vast majority of those components which exhibited strong multiplier behavior were located at the intersection of major functional areas

The results of the change structure were also depicted and considered through network graphs.

At an aggregate level, the change frequencies between areas of the program were calculated from the data, and compared to describe specific areas according to the multiplier, carrier, constant, absorber classification created by Eckert, Clarkson and Zanker and previously applied by de Weck and Suh. The results were evaluated regarding possible applications at an architectural design or program management level.

- Components at major functional intersections should be considered 'gray boxes', and be worked by a cross-functional team beginning in the design phase.

In addition, comparing the 'gray box' issue with consideration of the staff assoso ated with sets of change requests indicates that

- Large change networks typically have a low number of contributions from each of a large number of people, and strong change multipliers sit at the junction of functional areas, therefore it is likely that more effective cross functional communication would be required to reduce propagation.

Additionally, the relatively small amount of time available to attempt to process data of this extent was a significant constraint- this analysis has only scratched the surface of what this data may have to offer.

Hopefully these paths and more will be explored, leading to a better understanding of change propagation in the quest to understand and create ever larger and more complex systems.

References

[1] Eckert, Clarkson and Zanker. "Change and Customization in Complex Engineering Domains" Research in Engineering Design 15 (2004): 1-21.

[2] Terwiesch and Loch. "Managing the Process of Engineering Change Orders: the Case of the Climate Control System in Automobile Development" Journal of Production Innovation and Management 16 (1999): 160-172.

[3] Wright, IC. "A Review of Research into Engineering Change Management: Implications for Product Design" Design Studies 18 (1997): 33-42.

[4] Clarkson, Simons and Eckert. "Predicting Change Propagation in Complex Design" Transactions of the ASME 126 (2004): 788-797.

[5] de Weck and Suh. "Flexible Product Platforms: Framework and Case Study" Proceedings of IDETC/CIE 2006 ASME 2006 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, Submitted to Research in Engineering Design (2006).

[6] Keller, Eger, Eckert and Clarkson. "Visualising Change Propagation" 15th International Conference on Engineering Design (2005): 62-63.

[7] Jarratt, Eckert and Clarkson. "Pitfalls of Engineering Change: Change Practice During Complex Product Design" Advances in Design (2005): 413-424.

[8] Pikosz and Malmqvist. "A Comparative Study of Engineering Change Management in Three Swedish Engineering Companies" Proceedings of the DETC98 ASME Design Engineering Technical Conference (1998): 78-85.

[9] Huang and Mak. "Current Practices of Engineering Change Management in UK Manufacturing Industries" International Journal of Operations & Product Management 19(1) (1999): 21-37.

[10] Martin and Ishii. "Design for variety: developing standardized and modularized product platform architectures" Research in Engineering Design 13 (2002): 213-235.

[11] Earl, Eckert and Clarkson. "Design Change and Complexity" 2nd Workshop on Complexity in Design and Engineering (2005).

[12] Keller, Eckert & Clarkson. "Viewpoints and Views in Engineering Change Management" GIST Technical Report 1 (2005): 168-172.

[13] Jarratt, Eckert and Clarkson. "Development of a Product Model to Support Engineering Change Management" Proceedings of the TMCE 1 (2004): 331-342.

[14] Cohen and Fulton. "A Data Approach to Tracking and Evaluating Engineering Changes" Proceedings of the DETC98 ASME Design Engineering Technical Conference (1998): DETC98/EIM5682.

[15] Leveson, Nancy G. "Intent Specifications: An Approach to Building Human-Centered Specifications" Software Engineering 26-1 (2000): 15-35.

[16] Sterman, John D. Business Dynamics: Systems Thinking & Modeling for a Complex World. McGraw-Hill/Irwin, 2000.

[17] United States. DoD Directive 5015.2. Department of Defense Records Management Program 19 June, 2002.

[18] Eppinger et al. "A Model-Based Method for Organizing Tasks in Product Development" Research in Engineering Design 6 (1994): 1-13.

[19] Smaling and de Weck. "Assessing Risks and Opportunities of Technology Infusion in System Design" Systems Engineering 10(1) (2007): 1-25.

[20] Human Factors International Usability Newsletter http://www.humanfactors.com/downloads/usability-newsletter.asp (2007).

[21] Usability Professionals' Association Website http://www.upassoc.org/ (2007).

Appendix A: Data Analysis Detail

A.1 Example Perl Scripts

The following scripts are examples of the simple Perl scripts used to analyze the extraordinary amounts of data analyzed in this thesis. The data was stored in several tables within a single SQL database for rapid access, and thus SQL queries could be used to retrieve only the desired data in a relatively efficient manner.

The script below is an example of that used to extract consolidated information from the SQL database for import to Matlab for change network mapping. Variations on this script were used to extract other subsets of data for different analyses addressed throughout this thesis.

```
#!/usr/bin/perl
print "Content-type:text/html\n\n";
use DBI;

$username = 'root';$password = '';$database = 'thesis';$hostname = '127.0.0.1';
$dbh = DBI->connect("dbi:mysql:database=$database;" .
"host=$hostname;port=3307", $username, $password);

$SQL= "select cr_id from change_requests limit 1";

$Select = $dbh->prepare($SQL);
$Select->execute();

my $n = 41551;

my $i = 10001;

while ($i <= $n) {

  my $DATA="filehandle";
  my $FilePath=">data_records.$i";
  open(DATA,$FilePath);

  my $upperbound = $i+9999;
  print ("$i, $upperbound\n");
    my $CrQuery = "select cr_id, date_created, last_update, area_affected,
change_magnitude, submitted_by, assignee, stageorigin, defectreason, problem, solution,
completion from change_requests where cr_id >=$i and cr_id <=$upperbound order by
cr_id ";

    my $CrSelect = $dbh->prepare($CrQuery);
    $CrSelect->execute();
```

```
        while ($Row = $CrSelect -> fetchrow_hashref)
{
        print "$Row->{cr_id}\n";
        print DATA "$Row->{cr_id} \n";
        print DATA "$Row->{date_created} ";
        print DATA "$Row->{last_update} \n";
        print DATA "$Row->{area_affected} \n";
        print DATA "$Row->{change_magnitude} \n";

        $ParentQuery = "select parent from parent
                where cr_id =('$Row->{cr_id}')";
        $ParentSelect = $dbh->prepare($ParentQuery);
        $ParentSelect->execute();
        while ($ParentRow = $ParentSelect->fetchrow_hashref) {
                print DATA "$ParentRow->{parent} ";
                }
        print DATA "\n";

        $ChildQuery = "select child from children
                where cr_id =('$Row->{cr_id}')";
        $ChildSelect = $dbh->prepare($ChildQuery);
        $ChildSelect->execute();
        while ($ChildRow = $ChildSelect->fetchrow_hashref) {
                print DATA "$ChildRow->{child} ";
                }
        print DATA "\n";

        $SiblingQuery = "select sibling from siblings
                where cr_id =('$Row->{cr_id}') and sibling is not null";
        $SiblingSelect = $dbh->prepare($SiblingQuery);
        $SiblingSelect->execute();
        while ($SiblingRow = $SiblingSelect->fetchrow_hashref) {
                print DATA "$SiblingRow->{sibling} ";
                }
        print DATA "\n";

        print DATA "$Row->{submitted_by}\n";
        print DATA "$Row->{assignee} ";

        $AssigneeQuery = "select assignee from status_history
                where str_number =('$Row->{cr_id}')";
        $AssigneeSelect = $dbh->prepare($AssigneeQuery);
        $AssigneeSelect->execute();
        while ($AssigneeRow = $AssigneeSelect->fetchrow_hashref) {
                print DATA " $AssigneeRow->{assignee} ";
                }
```

```
        print DATA "\n";

## now pull engNNN adminNNN Engineer_NNN Admin_NNN out of text fields
## and print them separated by single spaces **
##
        while ($Row-
>{problem}=~m/(eng\d\d\d|admin\d\d\d|Engineer_\d\d\d|Admin_\d\d\d)/g){
        print DATA"$1 ";

}
#       print DATA"\n";

        while ($Row-
>{solution}=~m/(eng\d\d\d|admin\d\d\d|Engineer_\d\d\d|Admin_\d\d\d)/g){
        print DATA"$1 ";
}
        print DATA"\n";
        print DATA "$Row->{stageorigin}, ";
        print DATA "$Row->{defectreason} \n";
        print DATA "$Row->{severity} \n";
        print DATA "$Row->{completion} \n";


}
 close(DATA);
$i = $i+10000;
}


#!/usr/bin/perl
print "Content-type:text/html\n\n";
use DBI;

$username = 'root';$password = '';$database = 'thesis';$hostname = '127.0.0.1';
$dbh = DBI->connect("dbi:mysql:database=$database;" .
"host=$hostname;port=3307", $username, $password);

 my $DATA="filehandle";
 my $FilePath=">area_generation_monthly";
 open(DATA,$FilePath);

# For each area:
for ($i = 1; $i <= 46; $i++) {

# For each year:
 for ($j = 1; $j <= 9; $j++){
```

This script shows how data was extracted for the monthly values per area seen in Figure 30. A similar script was used to derive other information based on time period.

```
# January

    # Do SQL query of database here:
    # define your query
    my $CountQuery = "select count(*) as the_count from change_requests where
area_affected =($i) and date_created like('%JAN-Y$j')";

    # have the db prepare the query (CountQuery was just a string.
    # Prepare converts that string into a select object
    my $CountSelect = $dbh->prepare($CountQuery);

    # tell that object to execute against the db
    $CountSelect->execute();

    my $month_count1;

    # fetch the rows of the result (we know there will only be one row)
    while ($CountRow = $CountSelect->fetchrow_hashref) {
        $month_count1 = $CountRow->{the_count};
    }
      print DATA "$i $j $month_count1 ";
        print "$i $j $month_count1 ";

# February

    # Do SQL query of database here:
    # define your query
    my $CountQuery = "select count(*) as the_count from change_requests where
area_affected =($i) and date_created like('%FEB-Y$j') ";

    # have the db prepare the query (CountQuery was just a string.
    # Prepare converts that string into a select object
    my $CountSelect = $dbh->prepare($CountQuery);

    # tell that object to execute against the db
    $CountSelect->execute();

    my $month_count2;

    # fetch the rows of the result (we know there will only be one row)
    while ($CountRow = $CountSelect->fetchrow_hashref) {
        $month_count2 = $CountRow->{the_count};
    }
```

```
        print DATA "$month_count2 ";
          print "$month_count2 ";

# March

    # Do SQL query of database here:
    # define your query
    my $CountQuery = "select count(*) as the_count from change_requests where
area_affected =($i) and date_created like('%MAR-Y$j') ";

    # have the db prepare the query (CountQuery was just a string.
    # Prepare converts that string into a select object
    my $CountSelect = $dbh->prepare($CountQuery);

    # tell that object to execute against the db
    $CountSelect->execute();

    my $month_count3;

    # fetch the rows of the result (we know there will only be one row)
    while ($CountRow = $CountSelect->fetchrow_hashref) {
          $month_count3 = $CountRow->{the_count};
    }
       print DATA "$month_count3 ";
          print "$month_count3 ";

# April

    # Do SQL query of database here:
    # define your query
    my $CountQuery = "select count(*) as the_count from change_requests where
area_affected =($i) and date_created like('%APR-Y$j') ";

    # have the db prepare the query (CountQuery was just a string.
    # Prepare converts that string into a select object
    my $CountSelect = $dbh->prepare($CountQuery);

    # tell that object to execute against the db
    $CountSelect->execute();

    my $month_count4;

    # fetch the rows of the result (we know there will only be one row)
    while ($CountRow = $CountSelect->fetchrow_hashref) {
          $month_count4 = $CountRow->{the_count};
    }
```

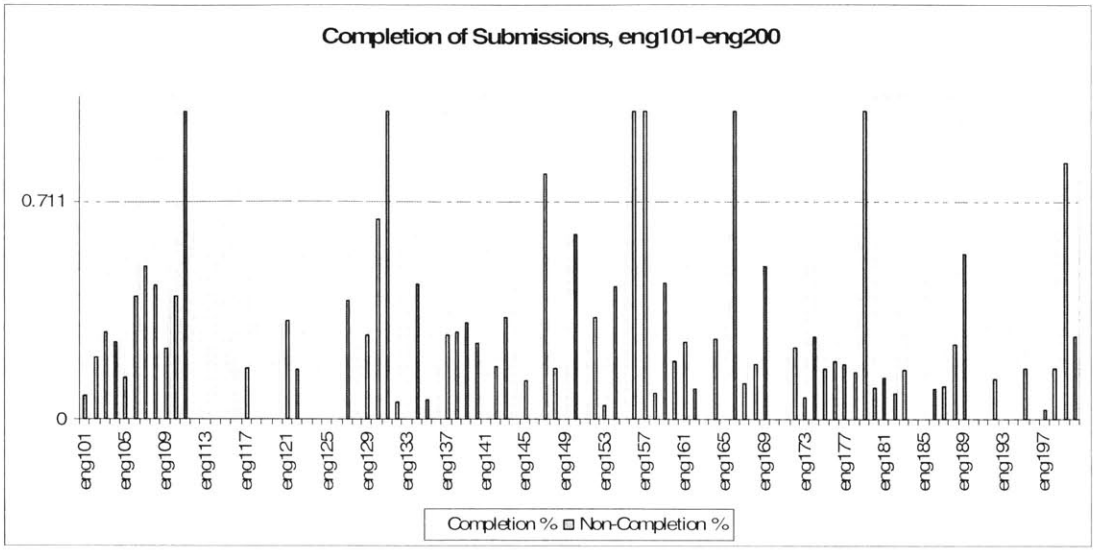
```
        print DATA "$month_count4 ";
          print "$month_count4 ";

# May

    # Do SQL query of database here:
    # define your query
    my $CountQuery = "select count(*) as the_count from change_requests where
area_affected =($i) and date_created like('%MAY-Y$j') ";

    # have the db prepare the query (CountQuery was just a string.
    # Prepare converts that string into a select object
    my $CountSelect = $dbh->prepare($CountQuery);

    # tell that object to execute against the db
    $CountSelect->execute();

    my $month_count5;

    # fetch the rows of the result (we know there will only be one row)
    while ($CountRow = $CountSelect->fetchrow_hashref) {
        $month_count5 = $CountRow->{the_count};
    }
       print DATA "$month_count5 ";
          print "$month_count5 ";

# June

    # Do SQL query of database here:
    # define your query
    my $CountQuery = "select count(*) as the_count from change_requests where
area_affected =($i) and date_created like('%JUN-Y$j') ";

    # have the db prepare the query (CountQuery was just a string.
    # Prepare converts that string into a select object
    my $CountSelect = $dbh->prepare($CountQuery);

    # tell that object to execute against the db
    $CountSelect->execute();

    my $month_count6;

    # fetch the rows of the result (we know there will only be one row)
    while ($CountRow = $CountSelect->fetchrow_hashref) {
        $month_count6 = $CountRow->{the_count};
    }
```

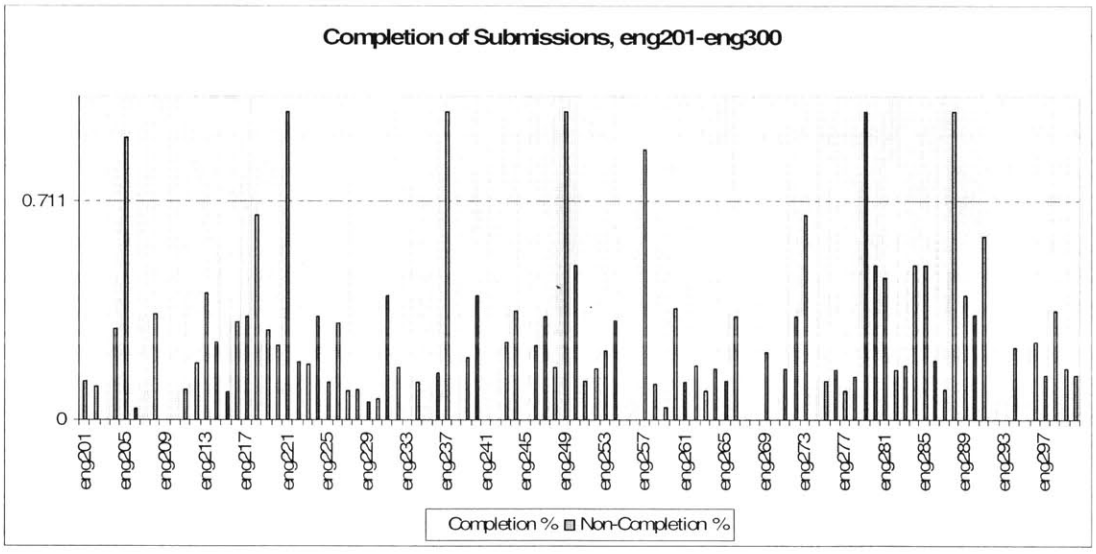
```
        print DATA "$month_count6 ";
           print "$month_count6 ";

# July

     # Do SQL query of database here:
     # define your query
     my $CountQuery = "select count(*) as the_count from change_requests where
area_affected =($i) and date_created like('%JUL-Y$j') ";

     # have the db prepare the query (CountQuery was just a string.
     # Prepare converts that string into a select object
     my $CountSelect = $dbh->prepare($CountQuery);

     # tell that object to execute against the db
     $CountSelect->execute();

     my $month_count7;

     # fetch the rows of the result (we know there will only be one row)
     while ($CountRow = $CountSelect->fetchrow_hashref) {
           $month_count7 = $CountRow->{the_count};
     }
        print DATA "$month_count7 ";
           print "$month_count7 ";

# August

     # Do SQL query of database here:
     # define your query
     my $CountQuery = "select count(*) as the_count from change_requests where
area_affected =($i) and date_created like('%AUG-Y$j') ";

     # have the db prepare the query (CountQuery was just a string.
     # Prepare converts that string into a select object
     my $CountSelect = $dbh->prepare($CountQuery);

     # tell that object to execute against the db
     $CountSelect->execute();

     my $month_count8;

     # fetch the rows of the result (we know there will only be one row)
     while ($CountRow = $CountSelect->fetchrow_hashref) {
           $month_count8 = $CountRow->{the_count};
     }
```

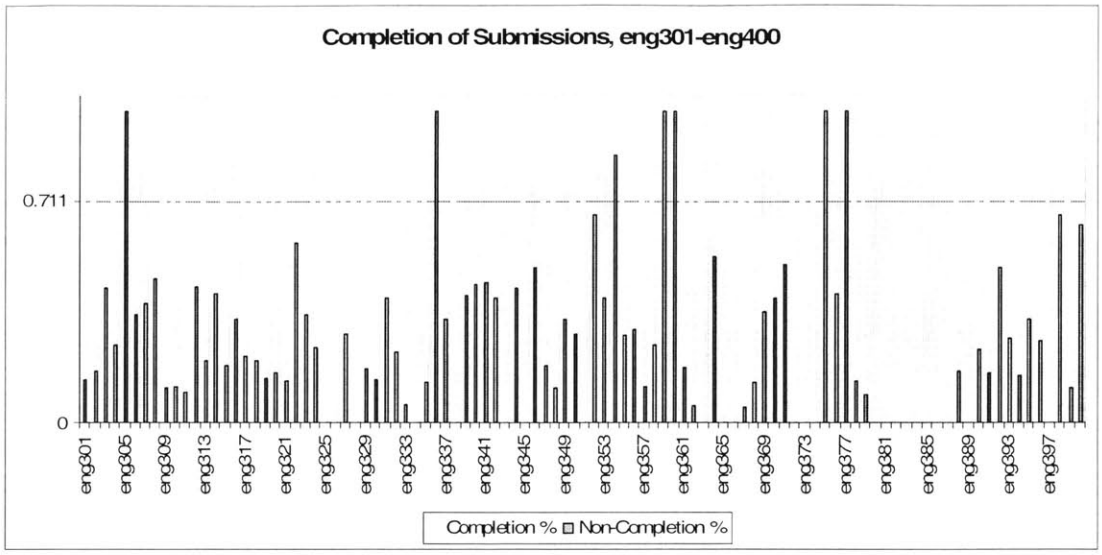
```
        print DATA "$month_count8 ";
          print "$month_count8 ";

# September

    # Do SQL query of database here:
    # define your query
    my $CountQuery = "select count(*) as the_count from change_requests where
area_affected =($i) and date_created like('%SEP-Y$j') ";

    # have the db prepare the query (CountQuery was just a string.
    # Prepare converts that string into a select object
    my $CountSelect = $dbh->prepare($CountQuery);

    # tell that object to execute against the db
    $CountSelect->execute();

    my $month_count9;

    # fetch the rows of the result (we know there will only be one row)
    while ($CountRow = $CountSelect->fetchrow_hashref) {
        $month_count9 = $CountRow->{the_count};
    }
        print DATA "$month_count9 ";
          print "$month_count9 ";
# October

    # Do SQL query of database here:
    # define your query
    my $CountQuery = "select count(*) as the_count from change_requests where
area_affected =($i) and date_created like('%OCT-Y$j') ";

    # have the db prepare the query (CountQuery was just a string.
    # Prepare converts that string into a select object
    my $CountSelect = $dbh->prepare($CountQuery);

    # tell that object to execute against the db
    $CountSelect->execute();

    my $month_count10;

    # fetch the rows of the result (we know there will only be one row)
    while ($CountRow = $CountSelect->fetchrow_hashref) {
        $month_count10 = $CountRow->{the_count};
    }
        print DATA "$month_count10 ";
```

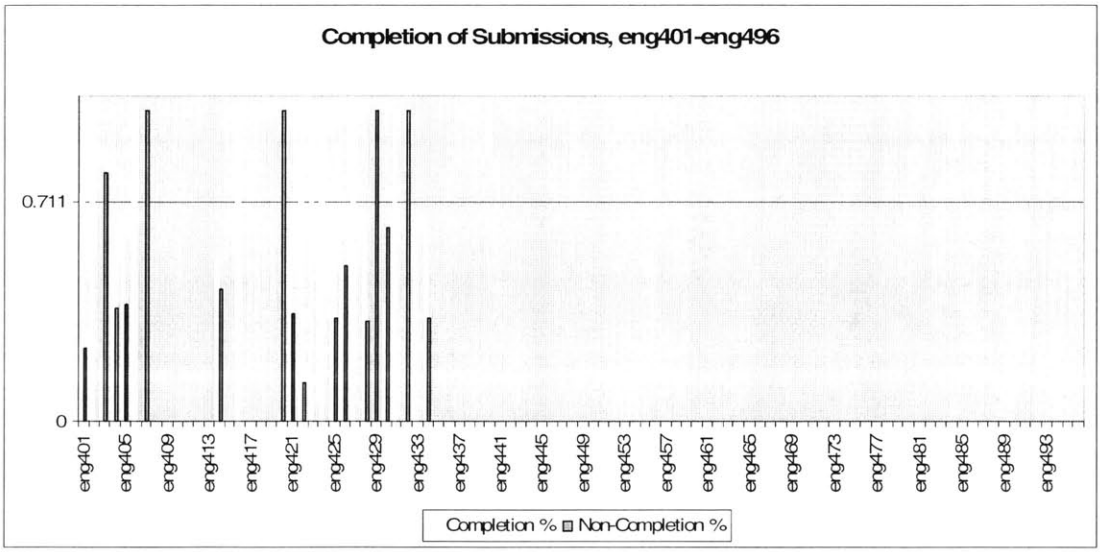
```
        print "$month_count10 ";
# November

    # Do SQL query of database here:
    # define your query
    my $CountQuery = "select count(*) as the_count from change_requests where
area_affected =($i) and date_created like('%NOV-Y$j') ";

    # have the db prepare the query (CountQuery was just a string.
    # Prepare converts that string into a select object
    my $CountSelect = $dbh->prepare($CountQuery);

    # tell that object to execute against the db
    $CountSelect->execute();

    my $month_count11;

    # fetch the rows of the result (we know there will only be one row)
    while ($CountRow = $CountSelect->fetchrow_hashref) {
        $month_count11 = $CountRow->{the_count};
    }
       print DATA "$month_count11 ";
        print "$month_count11 ";
# December

    # Do SQL query of database here:
    # define your query
    my $CountQuery = "select count(*) as the_count from change_requests where
area_affected =($i) and date_created like('%DEC-Y$j') ";

    # have the db prepare the query (CountQuery was just a string.
    # Prepare converts that string into a select object
    my $CountSelect = $dbh->prepare($CountQuery);

    # tell that object to execute against the db
    $CountSelect->execute();

    my $month_count12;

    # fetch the rows of the result (we know there will only be one row)
    while ($CountRow = $CountSelect->fetchrow_hashref) {
        $month_count12 = $CountRow->{the_count};
    }
       print DATA "$month_count12\n";
        print "$month_count12\n";
```

```
# Close the $j 'while'
    }

# Close the $i 'while'
    }

close(DATA);
```


Completion of Submissions, eng101-eng200
0.711
0
eng101
eng105
eng109
eng113
eng117
eng121
eng125
eng129
eng133
eng137
eng141
eng145
eng149
eng153
eng157
eng161
eng165
eng169
eng173
eng177
eng181
eng185
eng189
eng193
eng197
Completion % Non-Completion %

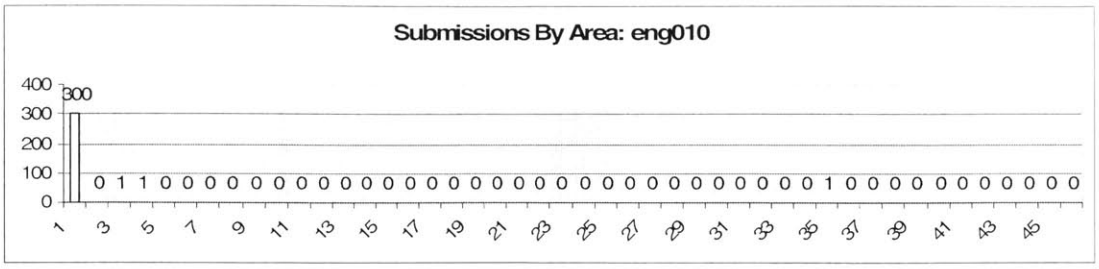

Completion of Submissions, eng201-eng300
0.711
0
eng201
eng205
eng209
eng213
eng217
eng221
eng225
eng229
eng233
eng237
eng241
eng245
eng249
eng253
eng257
eng261
eng265
eng269
eng273
eng277
eng281
eng285
eng289
eng293
eng297
Completion % Non-Completion %

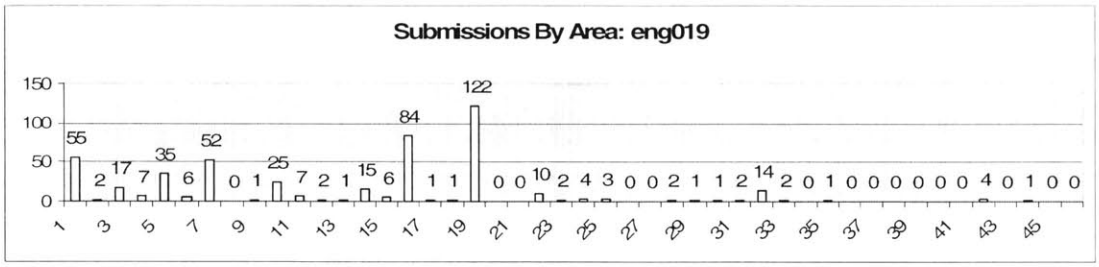

Completion of Submissions, eng301-eng400
0.711
0
eng301
eng305
eng309
eng313
eng317
eng321
eng325
eng329
eng333
eng337
eng341
eng345
eng349
eng353
eng357
eng361
eng365
eng369
eng373
eng377
eng381
eng385
eng389
eng393
eng397
Completion %
Non-Completion %


Completion of Submissions, eng401-eng496
0.711
0
eng401
eng405
eng409
eng413
eng417
eng421
eng425
eng429
eng433
eng437
eng441
eng445
eng449
eng453
eng457
eng461
eng465
eng469
eng473
eng477
eng481
eng485
eng489
eng493
Completion %
Non-Completion %

B.2 Examples of Difference in Individual Staff Work Patterns

The following graphs illustrate the extreme differences seen in individual work patterns as shown through the distribution of changes submitted against different areas by one individual throughout their time on the project.

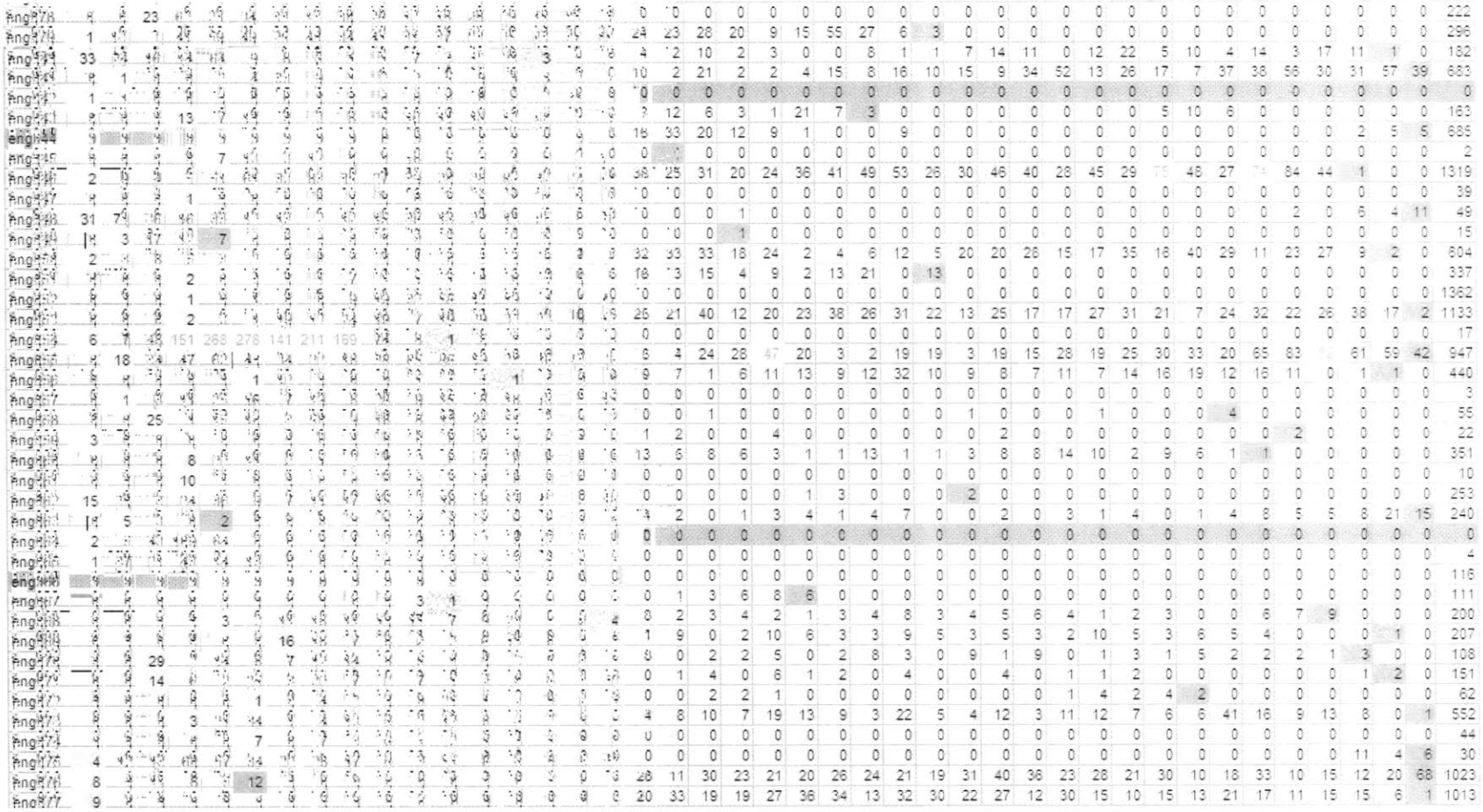


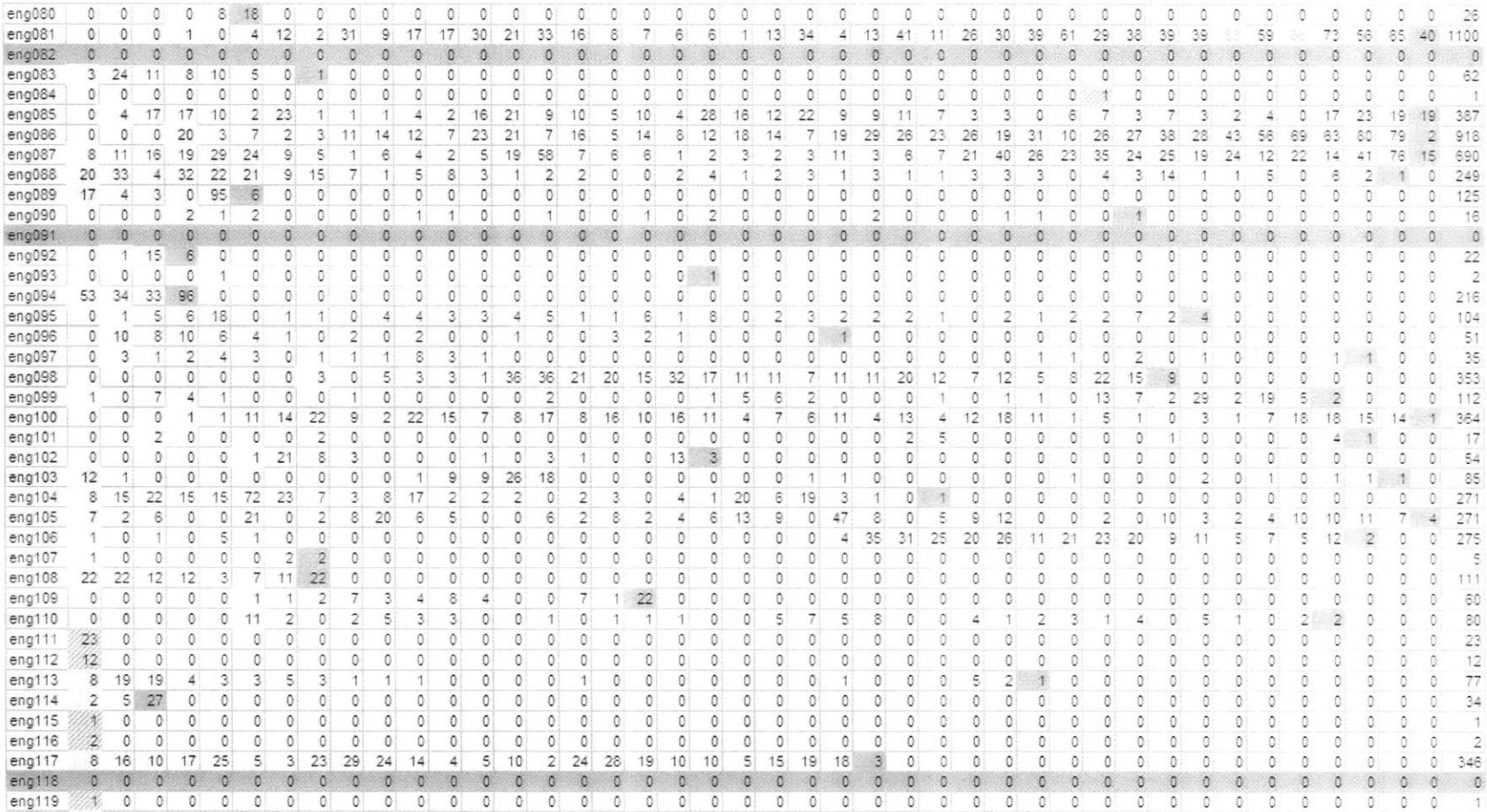


B.3 Individual Staff Submissions per Thousand Change Requests Written

The following pages detail the frequency of change request submissions by each engineer throughout the course of the program. In this case units of 1000 change requests are used as a proxy for duration (e.g. eng018 submitted 21 change requests out of the second 1000 written, and had ceased to contribute change requests to the program by the time the 13,000th change request was written).

	33	34	36	36	37	38	39	40	41	42	Total
	0	0	0	0	0	0	0	0	0	0	24
	0	0	0	0	0	0	0	0	0	0	7
	0	0	0	0	0	0	0	0	0	0	57
	0	0	0	0	0	0	0	0	0	0	5
	11	15	7	10	12	9	8	10	0	0	355
	0	3	5	6	5	21	11	0	0	0	145
	0	0	0	0	0	0	0	0	0	0	12
	13	0	0	1	0	0	0	0	0	0	790
	3	13	3	0	1	2	0	2	3	0	82
	4	10	7	7	16	8	13	5	1	0	757
	0	0	0	0	0	0	0	0	0	0	18
	0	0	0	0	0	0	0	0	0	0	0
	11	12	18	16	37	22	12	0	0	0	679
	1	1	0	0	0	0	0	0	2	1	45
	22	28	28	30	6	3	53	12	11	6	0	0	1121
	2	11	47	21	43	30	25	16	18	12	713
	7	28	13	40	26	16	30	28	40	39	758
	0	0	0	0	0	0	0	0	0	0	188
	48	51	57	6	11	12	22	11	41	8	1291
	0	0	0	0	0	0	0	0	1	0	34
	16	8	16	10	13	21	18	25	30	1	612
	3	14	3	0	3	11	0	24	17	20	355
	0	0	0	0	0	0	0	0	0	0	182
	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	1
	0	0	0	0	0	0	0	0	0	0	2
	0	0	0	0	0	0	0	0	0	0	0
	7	1	2	3	8	1	11	0	0	0	456
	0	0	0	0	0	0	0	0	0	0	0
	1	5	8	0	8	8	2	0	0	0	682
	1	0	5	2	0	1	0	0	0	0	186
	0	0	0	0	0	0	0	0	0	0	660
	0	0	0	0	0	0	0	0	2	5	60
	16	25	16	27	12	25	18	9	0	0	988
	0	0	0	0	0	0	0	0	0	0	9
	0	0	0	0	0	0	0	0	0	0	605
	0	0	0	0	0	2	0	2	21	2	8	86
	0	0	0	0	0	0	0	0	0	0	28
	0	0	0	0	0	0	0	0	0	0	17





eng120	1	10	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	15
eng121	1	0	12	32	44	37	14	57	49	76	48	23	27	48	37	36	36	45	28	34	39	26	34	31	28	38	35	70	37	32	28	32	35	40	57	43	22	33	16	9	63	0	1432
eng122	1	20	29	61	70	74	53	60	40	36	34	14	21	30	31	18	36	55	27	22	22	12	21	26	15	26	20	22	28	16	40	23	20	16	15	12	21	24	20	44	41	7	1223
eng123	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
eng124	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
eng125	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
eng126	0	2	0	0	3	1	2	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9
eng127	0	0	0	0	7	7	36	5	3	0	2	0	6	2	12	6	1	4	1	2	7	2	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	105
eng128	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4
eng129	1	28	35	22	18	13	26	1	15	9	10	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	184
eng130	10	16	19	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	46
eng131	12	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	13
eng132	0	2	11	38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	51
eng133	0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16
eng134	0	8	3	10	9	0	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	43
eng135	0	2	1	0	8	5	0	2	3	1	1	1	1	1	12	8	11	2	2	4	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	66
eng136	0	0	3	2	0	1	0	2	2	1	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	15
eng137	0	19	27	8	4	5	2	0	0	0	2	0	0	0	2	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	70
eng138	0	0	0	0	0	1	1	7	0	10	1	26	2	4	3	0	1	7	0	0	0	2	1	3	2	0	4	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	76
eng139	0	3	22	2	2	6	1	2	0	1	1	1	1	1	1	2	2	0	0	1	2	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	52
eng140	0	31	23	23	25	23	24	89	74	73	76	20	23	51	98	40	14	22	20	9	15	17	10	11	22	8	33	15	35	11	12	35	37	25	12	20	27	32	20	6	0	0	1161
eng141	6	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8
eng142	0	4	2	0	0	0	2	11	9	9	24	35	50	13	19	17	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	197
eng143	0	1	0	3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5
eng144	0	0	0	0	1	0	0	2	3	0	4	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11
eng145	0	0	0	0	14	44	7	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	66
eng146	0	1	0	0	2	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5
eng147	0	0	0	3	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5
eng148	0	0	0	6	3	9	14	6	13	39	58	29	11	7	39	104	42	90	17	54	35	34	24	33	54	23	12	15	23	31	20	0	0	0	0	0	0	0	0	0	0	0	845
eng149	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8
eng150	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5
eng151	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4
eng152	1	4	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8
eng153	18	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	34
eng154	1	5	9	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	17
eng155	0	11	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	19
eng156	0	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7
eng157	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng158	0	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	13
eng159	0	2	14	4	0	3	0	0	0	0	9	4	1	0	0	1	0	0	3	5	1	6	0	1	31	0	0	0	3	0	1	2	1	1	0	1	1	0	0	0	0	0	95

eng160	0	2	18	18	0	1	1	0	1	1	19	1	0	1	11	0	0	1	0	0	7	3	2	0	0	0	1	6	3	0	3	1	2	3	2	7	2	3	2	0	0	0	122
eng161	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4
eng162	0	0	17	13	22	3	13	10	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	80
eng163	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3
eng164	0	0	2	0	0	11	3	0	0	4	4	0	1	0	0	0	0	0	4	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	31
eng165	0	0	1	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4
eng166	0	0	0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8
eng167	0	0	0	18	1	17	0	23	6	0	0	1	0	1	1	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	71
eng168	0	2	1	30	27	31	73	34	36	41	17	31	76	39	137	131	70	108	68	32	8	4	55	61	66	128	57	63	48	36	10	0	0	0	0	0	0	0	0	0	0	0	1560
eng169	0	0	38	52	4	0	0	3	0	0	20	42	37	0	0	0	0	0	0	0	0	0	3	0	1	1	5	0	2	0	0	5	11	2	4	2	3	1	0	0	7	4	247
eng170	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	2
eng171	0	0	0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8
eng172	0	0	0	3	0	5	3	23	6	17	6	3	7	10	20	30	28	30	9	22	10	15	10	21	20	10	30	31	14	9	33	24	30	22	30	31	4	46	51	52	29	29	773
eng173	0	0	0	1	8	19	9	4	5	6	2	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	56
eng174	0	3	1	7	8	4	18	17	27	26	20	26	24	55	39	4	5	5	10	15	11	36	4	35	0	57	32	25	15	0	0	0	0	0	0	0	0	0	0	0	0	0	529
eng175	0	0	0	8	0	0	26	0	0	0	16	17	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	72
eng176	0	0	0	4	13	2	9	15	9	0	12	8	1	2	0	4	6	9	10	7	11	9	5	10	6	3	7	7	29	12	11	27	15	18	6	7	0	0	0	0	2	7	303
eng177	0	0	0	0	20	11	18	16	15	4	1	6	4	4	5	6	17	4	7	14	4	21	17	7	8	2	5	4	12	5	22	27	15	22	16	29	25	42	49	72	77	19	652
eng178	0	0	0	1	21	28	7	62	26	23	13	35	64	69	39	19	7	13	18	2	6	19	15	8	8	7	29	17	13	6	36	15	13	34	7	4	17	2	30	0	0	0	733
eng179	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng180	0	1	0	0	4	6	2	0	1	0	0	0	3	5	0	0	2	2	6	2	0	0	0	2	0	1	7	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	46
eng181	1	4	1	6	16	13	2	3	0	2	10	18	11	9	13	0	6	6	35	4	5	6	39	13	1	5	1	5	4	4	10	1	3	1	7	5	0	0	0	0	0	0	270
eng182	0	0	0	0	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12
eng183	0	1	0	0	63	11	19	14	18	8	36	29	26	74	51	33	47	32	35	23	56	27	47	45	57	54	32	52	51	27	61	64	49	67	54	32	74	41	38	[illegible]	93	23	[illegible]
eng184	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3
eng185	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	3
eng186	0	0	0	24	7	4	1	8	6	5	8	8	5	1	1	2	1	14	3	9	7	3	3	11	12	8	13	1	9	13	1	3	0	0	0	0	0	0	0	0	0	0	191
eng187	0	0	0	0	1	15	56	48	28	2	1	11	17	5	24	15	16	33	31	22	4	13	29	7	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	384
eng188	0	0	0	0	3	18	6	21	2	0	7	0	0	0	5	1	0	6	4	0	3	5	4	7	1	1	4	5	0	0	5	22	10	8	0	5	0	1	1	0	0	0	155
eng189	0	0	0	1	18	25	2	12	20	14	1	13	0	1	3	7	3	2	0	1	2	4	6	10	11	6	1	0	19	10	3	1	8	1	2	1	0	0	0	0	0	0	208
eng190	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng191	0	2	1	2	1	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12
eng192	0	0	0	16	7	3	10	9	34	7	3	4	12	49	29	13	9	3	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	210
eng193	0	0	0	0	1	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6
eng194	0	0	0	0	0	2	0	2	0	2	2	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9
eng195	0	1	2	4	4	4	7	13	19	42	15	31	7	6	23	3	4	2	9	3	3	13	12	2	8	11	9	6	5	3	9	4	9	5	4	5	23	9	23	35	66	35	498
eng196	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	3
eng197	0	0	0	0	0	8	28	10	14	0	1	2	3	0	0	0	5	14	27	9	25	11	28	6	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	198
eng198	0	0	0	5	5	10	9	12	14	23	15	5	14	7	5	12	6	4	18	6	13	6	1	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	192
eng199	0	0	0	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6

eng200	0	0	0	0	0	2	14	36	12	5	1	10	11	9	5	5	6	5	8	17	8	26	17	5	6	1	3	9	17	18	0	9	5	13	24	15	25	34	38	29	31	4	483
eng201	0	0	0	1	1	0	15	10	11	1	17	10	11	14	18	25	24	38	15	15	19	21	33	15	20	43	20	40	23	32	30	60	42	31	42	36	36	43	53	59	17	1	1042
eng202	0	0	0	0	4	7	14	19	16	28	1	31	16	33	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	177
eng203	0	0	44	14	13	32	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	109
eng204	0	1	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	2	3	2	22	4	12	3	0	0	2	7	4	1	3	0	0	0	0	0	0	0	0	0	0	0	68
eng205	0	0	0	0	0	0	5	0	0	0	0	11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	18
eng206	0	0	0	0	0	0	6	1	11	5	9	4	8	2	1	0	1	6	0	3	5	16	19	16	35	14	25	28	28	8	10	8	7	5	0	0	0	0	0	0	0	0	281
eng207	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
eng208	0	0	0	0	1	8	19	14	16	15	0	1	0	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	77
eng209	0	0	0	0	0	0	18	7	7	0	0	0	4	48	5	1	0	20	14	5	2	3	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	137
eng210	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
eng211	0	0	0	0	1	0	7	7	5	3	7	0	8	0	2	1	10	0	3	3	5	1	4	3	2	3	1	8	4	8	5	5	9	4	11	5	3	2	0	0	0	1	141
eng212	0	0	0	0	0	0	1	10	21	25	8	13	46	20	15	21	2	0	19	2	45	35	8	10	14	24	53	8	6	38	7	15	16	14	9	14	17	21	31	50	70	21	729
eng213	0	0	0	0	0	0	0	20	7	1	0	0	0	0	0	0	2	5	9	6	9	2	6	2	0	0	2	22	1	0	0	2	2	13	5	2	0	11	6	23	2	0	160
eng214	0	0	0	0	0	0	0	3	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7
eng215	0	0	0	0	0	0	0	16	19	18	6	7	7	20	3	10	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	113
eng216	0	0	1	1	2	0	3	16	20	33	41	44	53	21	54	17	12	9	37	13	7	23	10	15	1	3	3	4	10	15	7	22	22	30	24	18	27	22	6	0	0	0	646
eng217	0	0	0	0	0	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6
eng218	0	0	0	0	0	0	0	5	1	0	0	7	0	2	0	0	0	4	0	0	0	0	2	1	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	25
eng219	0	0	0	0	0	1	6	1	10	27	7	16	1	6	0	7	23	3	8	5	14	2	0	5	0	1	1	7	9	2	0	2	2	0	0	0	0	0	0	0	0	0	166
eng220	0	0	0	0	1	7	1	4	0	6	8	0	0	0	34	34	39	6	0	0	4	4	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	149
eng221	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3
eng222	0	0	0	2	1	4	0	13	16	16	14	11	16	16	17	35	21	16	26	5	6	23	12	7	14	33	16	29	11	34	20	4	16	17	3	16	8	13	17	3	0	0	531
eng223	0	0	0	0	0	0	0	1	0	0	0	0	9	0	0	0	3	0	0	2	3	4	1	4	0	5	5	9	3	3	0	6	3	7	3	3	4	4	1	0	0	0	83
eng224	0	0	0	0	0	0	0	3	7	11	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	25
eng225	0	0	0	0	0	0	0	2	1	11	1	3	3	1	3	2	5	13	0	5	6	0	16	3	8	0	7	2	3	0	0	0	0	0	0	0	0	0	0	0	0	0	95
eng226	0	0	0	0	14	0	0	1	0	2	0	2	0	0	0	0	0	4	3	2	5	1	3	1	1	0	0	2	0	0	0	0	2	0	0	0	0	0	0	0	0	0	43
eng227	0	0	0	1	2	0	0	6	8	9	1	13	14	20	8	2	1	9	9	3	10	21	8	2	24	8	18	14	28	22	20	41	26	6	16	16	13	9	13	1	0	0	422
eng228	0	0	0	1	2	0	0	3	15	2	2	9	5	5	6	0	3	3	3	0	2	6	3	1	0	0	4	7	6	0	1	1	6	8	6	4	2	8	3	4	2	5	138
eng229	0	0	0	0	0	0	0	0	1	1	1	4	1	3	4	2	9	8	2	0	0	1	5	2	2	2	1	0	1	2	7	2	4	2	4	2	9	7	10	12	19	8	138
eng230	0	0	0	0	0	0	0	1	20	10	7	0	1	10	1	3	3	3	3	2	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	66
eng231	0	0	0	0	1	0	0	1	4	39	4	2	8	12	9	8	16	14	4	7	1	13	6	2	10	13	4	3	3	13	12	21	10	24	3	9	18	13	11	20	20	45	403
eng232	0	0	0	0	1	0	1	2	9	15	46	28	18	14	10	33	22	25	21	45	29	38	27	27	24	18	6	6	36	21	26	22	49	35	23	24	28	24	27	40	0	0	820
eng233	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4
eng234	0	0	0	1	2	1	0	7	12	11	1	7	3	2	4	13	8	12	4	15	3	9	3	5	5	5	4	13	6	7	20	8	10	6	3	3	5	1	0	0	0	0	219
eng235	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
eng236	0	0	0	0	0	0	0	0	25	7	10	8	17	2	8	1	9	35	32	36	14	20	15	15	24	21	13	12	27	16	17	20	10	11	31	31	35	28	27	28	28	14	645
eng237	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng238	0	0	0	0	0	0	0	0	1	0	0	0	34	0	0	0	0	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	38
eng239	0	0	0	0	0	0	1	1	6	12	0	0	12	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	42

eng240	0	0	0	0	0	0	0	0	2	2	0	0	0	0	0	0	0	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7
eng241	0	0	0	0	0	0	0	0	9	1	10	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	23
eng242	0	0	0	0	0	1	6	1	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12
eng243	0	0	0	0	0	0	0	0	3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4
eng244	0	0	0	0	0	0	0	1	6	17	4	10	0	7	6	6	4	11	6	2	1	7	13	11	0	4	10	11	9	22	14	22	15	22	9	14	13	23	18	5	1	2	326
eng245	0	0	0	0	0	0	0	1	2	22	4	4	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	38
eng246	0	0	0	0	0	0	0	0	0	2	0	22	4	1	0	1	0	1	0	5	0	1	0	0	0	1	0	0	0	1	1	0	0	0	0	0	1	0	0	2	12	15	70
eng247	0	0	0	0	0	0	1	0	13	18	2	19	32	16	34	5	3	5	15	22	37	30	0	0	25	17	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	298
eng248	0	0	0	0	1	0	2	3	8	1	8	13	24	14	17	3	15	11	8	15	11	18	8	5	9	10	4	11	28	15	5	5	14	15	19	21	12	7	11	0	0	0	371
eng249	0	0	0	0	0	0	0	0	0	2	0	6	0	0	0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16
eng250	0	0	0	0	0	0	0	1	0	1	8	83	0	23	0	16	0	0	0	0	1	0	0	10	0	3	12	0	3	1	0	10	5	4	11	7	6	11	5	1	0	0	222
eng251	0	0	0	0	0	0	0	1	0	5	7	4	0	4	4	3	1	3	3	28	10	22	14	5	8	19	16	23	13	15	5	30	31	31	10	3	0	0	0	0	0	0	0	0	318
eng252	0	0	0	0	0	0	0	0	0	2	1	0	15	0	0	2	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	25
eng253	0	1	0	2	0	1	0	0	0	1	32	5	0	0	11	1	1	0	3	13	6	3	13	13	21	1	5	5	5	8	2	6	5	5	7	0	2	1	3	5	4	0	191
eng254	0	0	0	0	0	0	0	0	0	0	5	1	1	2	3	2	0	12	7	1	0	6	4	6	7	3	6	7	12	12	13	9	9	15	28	34	21	18	9	1	0	0	254
eng255	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
eng256	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
eng257	0	0	0	0	0	0	0	0	0	0	2	4	1	0	0	0	0	0	0	6	5	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	27
eng258	0	0	0	0	0	0	2	0	5	1	8	2	5	0	0	1	2	17	7	12	14	15	12	24	24	25	19	6	16	14	5	13	4	5	19	2	0	0	0	0	0	0	279
eng259	0	0	0	0	0	0	0	0	0	0	6	10	6	20	4	0	9	8	0	12	7	12	4	12	6	11	10	28	18	7	6	12	10	7	2	0	0	0	5	0	0	0	232
eng260	0	0	0	0	0	0	0	0	1	0	1	11	6	1	1	14	7	24	30	14	4	15	1	34	17	25	11	7	16	5	10	5	7	17	7	26	13	38	26	27	71	14	506
eng261	0	0	0	0	0	0	2	0	1	0	12	2	2	15	17	2	3	7	11	9	8	1	5	0	8	3	20	12	8	8	7	11	5	10	5	18	7	14	6	3	0	0	242
eng262	0	0	0	0	0	0	0	0	0	0	4	12	12	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	30
eng263	0	0	0	0	0	0	0	6	0	0	7	26	13	9	3	9	1	10	0	1	8	2	6	14	14	11	15	6	6	0	13	17	19	4	8	9	17	15	13	33	23	3	341
eng264	0	0	0	0	0	0	0	0	9	0	24	1	11	29	23	19	27	16	18	29	10	21	16	17	32	25	29	45	14	30	19	16	9	18	13	8	1	1	8	0	0	0	538
eng265	0	0	0	0	0	0	0	0	0	0	6	11	0	0	9	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	29
eng266	0	0	0	0	0	0	0	0	0	0	10	3	0	0	0	0	2	1	0	12	15	6	2	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	67
eng267	0	0	0	0	0	0	0	0	3	1	1	14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	19
eng268	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6
eng269	0	0	0	0	0	1	1	3	3	2	6	1	0	1	5	4	5	10	3	1	2	0	5	0	1	3	1	2	5	7	3	3	6	5	5	1	5	6	9	12	9	13	149
eng270	0	0	0	0	0	17	0	0	0	0	2	0	2	3	2	3	1	0	3	3	2	3	2	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	45
eng271	0	0	1	0	1	0	0	4	7	19	13	26	31	15	6	5	14	7	6	6	3	2	10	4	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	182
eng272	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3
eng273	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4
eng274	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng275	0	0	0	1	0	1	0	2	1	12	15	15	13	25	20	9	46	32	12	70	26	37	26	36	26	22	16	15	11	16	27	20	25	13	14	23	19	17	21	19	33	3	739
eng276	0	0	0	0	0	0	0	0	0	0	6	2	5	14	2	12	9	15	12	26	6	5	13	32	15	17	18	9	13	10	7	2	0	0	0	0	0	0	0	0	0	0	250
eng277	0	0	0	0	0	0	0	0	0	0	1	4	3	2	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	26
eng278	0	0	0	0	0	0	0	0	12	0	18	1	8	2	0	3	3	19	14	12	37	4	12	24	19	20	12	5	16	15	17	11	14	14	9	14	25	14	16	5	0	0	395
eng279	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4

eng280	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12
eng281	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	4	13	1	0	4	5	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	36
eng282	0	0	0	0	0	0	0	0	0	0	2	11	16	16	18	13	30	15	4	8	16	10	3	10	5	5	5	24	3	19	14	10	16	13	9	9	0	0	0	0	0	0	304
eng283	0	0	0	0	0	0	0	1	0	0	0	4	11	0	1	2	0	2	16	13	2	9	11	25	14	11	3	4	0	5	10	0	0	0	0	0	0	0	0	0	0	0	144
eng284	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
eng285	0	0	0	2	0	0	1	4	0	15	0	0	1	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	26
eng286	0	0	0	0	0	0	0	0	0	0	0	7	4	0	4	0	3	4	1	3	3	2	0	1	0	5	2	0	4	0	5	8	8	1	9	15	28	0	6	58	30	37	248
eng287	0	0	0	0	0	0	0	0	0	0	0	0	0	2	1	2	1	8	0	1	0	0	3	1	0	17	3	13	5	18	62	24	2	13	4	12	27	43	19	0	0	0	281
eng288	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng289	0	0	0	0	1	0	1	0	0	0	1	0	3	2	0	1	1	5	0	0	1	1	1	0	1	1	0	0	1	0	3	2	0	0	0	0	0	0	0	0	0	0	26
eng290	0	0	0	0	0	0	0	0	0	0	0	14	0	5	7	6	7	7	2	2	4	2	6	1	0	0	1	0	3	4	5	1	0	6	5	0	4	17	3	0	0	0	112
eng291	0	0	0	0	0	0	0	0	0	0	0	8	2	3	7	0	6	1	3	30	27	8	1	8	44	3	1	5	1	3	1	8	0	3	6	6	5	3	3	0	0	0	196
eng292	0	0	0	0	0	0	0	0	6	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	8
eng293	0	0	0	0	0	0	0	0	0	0	0	0	0	1	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5
eng294	0	0	0	0	0	0	0	0	0	0	0	0	0	6	11	0	2	13	6	3	0	15	0	0	0	0	10	20	17	19	6	12	22	14	11	20	20	10	0	0	0	0	237
eng295	0	0	0	0	0	0	0	0	0	0	0	0	0	6	7	4	15	2	3	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	39
eng296	0	0	0	0	0	1	0	0	0	0	0	2	0	5	6	18	16	5	1	6	10	14	4	2	8	16	6	2	8	16	7	17	8	11	10	41	6	4	23	0	0	0	273
eng297	0	0	0	0	0	0	0	0	0	0	0	0	0	9	3	22	4	0	10	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	51
eng298	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	2	0	0	0	2	1	0	0	0	0	0	1	0	1	1	3	1	2	0	0	0	18
eng299	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	5	11	2	6	16	3	2	8	2	6	13	14	12	18	29	25	49	43	43	19	23	15	34	24	46	68	13	552
eng300	0	0	0	0	0	0	0	0	0	0	1	0	0	17	4	4	26	28	4	14	0	14	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	121
eng301	0	0	0	0	0	0	0	0	0	0	0	0	0	4	10	27	21	4	5	27	4	19	23	12	4	19	51	54	15	28	17	31	17	22	1	12	13	15	22	13	30	29	549
eng302	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	4	44	20	8	2	27	17	27	37	43	25	36	24	35	24	27	13	19	17	7	22	27	22	23	26	52	25	657
eng303	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	5	0	0	4	1	4	4	0	0	2	2	2	3	3	3	5	2	3	5	0	5	1	4	1	3	0	0	64
eng304	0	0	0	0	0	1	0	0	0	0	1	0	0	2	5	0	2	0	17	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	30
eng305	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	1	11	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	26
eng306	0	0	0	1	1	0	0	0	0	3	1	2	3	12	22	10	1	44	6	12	9	5	11	14	1	3	7	1	14	11	3	8	7	4	7	14	0	0	0	0	0	0	237
eng307	0	0	0	0	1	1	0	0	0	0	0	1	0	1	7	2	2	1	1	0	1	0	44	2	2	0	0	0	64	2	0	0	12	14	3	25	11	4	25	0	0	0	226
eng308	0	0	0	0	0	0	0	0	0	0	0	0	0	14	4	1	4	2	0	10	26	5	3	1	11	0	47	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	137
eng309	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	6	78	16	36	10	22	23	23	18	25	13	11	5	23	10	19	12	30	45	15	38	12	16	0	0	0	0	509
eng310	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	43	9	20	9	7	9	13	19	33	19	16	16	17	8	12	22	19	18	29	19	10	13	7	14	21	0	0	424
eng311	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	8	15	13	10	26	18	17	11	21	25	25	7	19	22	25	13	41	22	24	5	12	15	6	16	27	25	30	500
eng312	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	8	1	3	8	3	2	3	25	23	14	1	12	16	11	16	18	0	4	17	29	20	18	12	19	36	15	339
eng313	0	0	0	0	0	0	0	0	1	1	3	3	3	13	14	13	2	1	6	3	1	0	0	5	3	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	75
eng314	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	9	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	17
eng315	0	0	0	0	0	0	0	0	0	0	0	1	2	0	1	2	4	3	10	12	16	26	5	4	3	4	8	0	9	10	8	9	9	4	12	5	12	22	8	26	13	0	248
eng316	0	0	0	0	0	0	1	0	0	1	2	2	0	1	9	54	9	19	46	23	21	26	16	31	12	6	9	17	18	19	22	3	2	9	8	22	8	2	4	31	15	2	470
eng317	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	6	4	3	1	15	7	36	34	23	6	8	17	6	4	25	2	2	19	5	6	9	25	9	1	0	0	0	277
eng318	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	6	16	0	4	3	2	3	6	0	3	8	9	13	5	8	5	7	16	7	13	8	10	6	11	7	0	184
eng319	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	6	12	0	2	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	28

eng320	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	2	10	8	6	24	12	10	35	32	20	10	5	3	12	13	11	11	22	8	12	14	12	13	21	27	12	28	396
eng321	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	14	14	11	10	3	3	1	3	9	11	18	7	22	22	38	44	29	41	26	33	24	23	35	52	88	81	666
eng322	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9	6	4	3	6	8	9	5	3	6	1	6	8	4	2	4	0	5	2	10	6	3	0	0	0	0	110
eng323	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	2	2	0	0	2	1	6	2	1	0	1	2	1	1	2	2	3	2	1	1	2	5	6	9	3	61
eng324	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	1	2	5	1	0	0	0	2	3	0	16	8	1	12	7	4	4	0	3	3	3	0	16	0	0	94
eng325	0	0	0	0	0	0	0	1	0	0	0	0	2	0	7	3	2	0	72	3	27	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	125
eng326	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng327	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	14	7	14	2	1	0	0	7	1	1	7	1	23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	78
eng328	0	0	0	0	0	1	0	0	0	0	1	1	1	1	0	2	2	13	1	3	5	0	0	1	0	1	2	0	0	5	4	2	1	0	0	4	1	1	0	0	0	1	54
eng329	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	23	2	25	2	2	13	5	21	7	22	7	22	26	12	13	19	22	15	7	16	11	14	0	0	0	312
eng330	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	5	6	8	7	8	16	13	5	2	4	9	13	18	5	8	10	3	13	9	6	11	5	5	40	31	38	301
eng331	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	1	2	4	0	2	0	2	0	3	0	0	0	2	0	1	5	1	0	27
eng332	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	9	10	5	9	7	6	12	4	2	4	7	8	7	7	28	8	6	5	5	13	7	15	9	0	1	195
eng333	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	4	14	12	11	13	11	2	4	11	8	5	22	23	41	30	39	44	30	20	26	4	0	0	376
eng334	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	2	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5
eng335	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	5	0	0	19	3	31	16	2	28	5	1	2	9	1	9	9	18	7	18	6	19	33	31	27	17	0	0	318
eng336	0	0	0	1	0	2	0	0	2	0	0	0	1	3	1	0	0	0	0	2	13	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	26
eng337	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	2	0	0	0	2	0	0	1	0	0	1	0	1	1	0	0	0	0	0	0	0	11
eng338	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	2
eng339	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	41	4	0	10	0	0	0	0	0	0	0	0	0	0	6	2	0	0	0	0	1	0	0	64
eng340	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	10	2	0	1	5	6	7	9	1	12	3	11	11	16	3	4	1	6	19	10	31	3	0	172
eng341	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	8	12	8	4	14	3	5	6	0	1	4	5	5	7	6	1	0	0	0	0	0	93
eng342	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	1	0	1	2	5	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	13
eng343	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	3	4	3	0	0	0	0	0	1	0	0	0	0	0	0	0	0	15
eng344	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	7	5	9	1	3	12	5	12	4	2	6	2	0	0	0	0	0	0	0	0	0	73
eng345	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	4	0	2	0	0	0	0	0	0	0	9
eng346	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	4	1	0	0	0	0	0	0	0	0	0	1	2	4	2	0	0	0	0	0	0	0	16
eng347	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	5	4	5	0	0	2	29	7	0	38	36	6	6	20	37	0	0	0	0	0	0	0	197
eng348	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	2	3	1	0	3	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	19
eng349	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	3
eng350	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0	2	2	2	8	6	6	20	20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	69
eng351	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	3	0	0	1	0	4	2	3	0	0	0	0	2	0	0	1	1	3	1	0	23
eng352	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3
eng353	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	5
eng354	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	6	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	15
eng355	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9	22	15	14	6	28	5	5	6	6	6	7	6	5	6	5	9	9	0	0	0	169
eng356	0	0	0	0	0	0	0	0	0	0	7	0	0	1	3	0	2	1	0	0	9	2	1	18	13	0	2	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	68
eng357	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	3	0	1	2	1	2	0	0	3	3	5	2	3	3	4	3	4	15	57	114
eng358	0	0	0	0	0	0	0	0	1	2	0	0	0	2	3	10	7	4	6	30	5	7	3	8	9	11	9	11	9	7	9	15	11	8	16	13	13	14	17	34	0	0	292
eng359	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3

eng360	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4
eng361	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	2	12	5	3	2	0	1	2	6	1	0	1	3	0	0	0	40
eng362	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	22
eng363	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng364	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	4	1	0	1	0	0	1	0	0	10	0	0	0	1	1	3	2	6	31
eng365	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
eng366	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng367	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	4	1	0	4	0	4	0	2	5	2	2	0	2	0	4	1	7	0	0	40
eng368	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	0	1	3	7	15	2	0	1	6	2	2	10	5	11	3	6	12	16	26	30	12	173
eng369	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	2	0	0	0	1	0	2	1	0	0	3	14	13	0	0	0	0	38
eng370	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	8
eng371	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	1	1	2	1	6	6	11	2	2	6	5	7	12	8	0	0	0	73
eng372	0	0	0	0	2	1	0	2	1	1	0	0	0	0	2	0	0	1	0	0	3	5	3	4	2	4	2	24	2	0	1	2	3	0	5	0	0	0	0	0	0	0	70
eng373	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	3	6	12	11	14	11	6	12	6	7	22	16	13	12	7	0	2	7	4	2	0	174
eng374	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	3
eng375	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng376	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	5	1	1	13	1	28	16	3	32	22	9	21	2	0	0	0	0	0	155
eng377	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
eng378	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	7	0	1	0	8	0	0	1	3	3	0	7	4	6	5	4	0	0	0	0	59
eng379	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	1	5	0	14	5	4	21	23	26	15	14	16	15	20	16	3	1	202
eng380	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng381	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng382	0	1	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8
eng383	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
eng384	0	0	15	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	25
eng385	0	0	0	1	0	0	0	2	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4
eng386	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3
eng387	0	0	0	0	0	1	5	0	4	7	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	33
eng388	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	1	0	21	3	3	4	6	12	9	21	22	47	35	188
eng389	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6
eng390	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	4	2	6	11	25	18	79	28	30	34	9	16	12	42	52	73	73	32	49	10	0	606
eng391	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1	0	0	2	2	2	13	9	3	7	22	25	20	20	36	47	42	0	254
eng392	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	1	1	6	4	9	0	0	0	1	10	5	2	1	3	3	7	4	7	0	0	68
eng393	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	4	3	0	0	3	0	0	0	0	0	0	0	0	0	0	14
eng394	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	1	2	3	5	2	5	8	4	3	3	0	5	5	3	0	0	0	51
eng395	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	3	0	0	0	0	2	0	1	9
eng396	0	0	0	1	0	0	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	1	0	1	0	0	1	0	6	0	6	4	4	0	29
eng397	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	2	0	0	2	0	0	0	0	0	0	0	6
eng398	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	1	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	5
eng399	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	11	0	0	0	0	0	1	0	1	0	8	9	15	9	14	21	15	4	0	0	0	0	0	0	0	0	0	113

eng400	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	2	2	8	0	1	0	1	0	0	0	0	0	0	0	0	0	15
eng401	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	14	8	42	54	13	16	26	48	20	33	57	46	46	31	15	477
eng402	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	3
eng403	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	4	0	0	0	0	0	0	0	0	0	0	0	0	5
eng404	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	4	2	6	1	1	0	2	1	0	1	19
eng405	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	1	1	1	1	0	2	4	0	6	14	3	0	0	37
eng406	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12	4	4	7	0	3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	31
eng407	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	6	2	0	0	0	0	0	0	0	0	0	10
eng408	0	0	0	1	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4
eng409	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
eng410	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	3	9	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	18
eng411	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng412	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
eng413	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	2	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8
eng414	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	3	0	4	7	5	5	6	13	10	17	12	2	86
eng415	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	4	6
eng416	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	7	12	6	10	9	31	9	2	0	0	0	0	0	0	91
eng417	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
eng418	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	1	1	0	1	0	1	0	0	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	9
eng419	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	2	0	0	0	0	0	0	0	0	0	0	0	5
eng420	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1
eng421	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	4	18	7	3	4	4	4	2	0	0	0	53
eng422	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	3	3	2	0	0	5	2	8	0	0	24
eng423	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
eng424	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
eng425	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	1	1	0	0	0	0	0	4
eng426	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	2
eng427	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
eng428	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	16	18	0	0	0	44
eng429	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	5	0	6
eng430	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12	3	15
eng431	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
eng432	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
eng433	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	11
eng434	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	13	14
eng435	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	4
eng436	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
eng437	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
eng438	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	13
eng439	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10

eng440	14	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	15
eng441	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng442	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng443	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng444	0	8	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	14
eng445	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7
eng446	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng447	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng448	0	0	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10
eng449	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
eng450	0	0	0	1	1	1	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5
eng451	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	2
eng452	0	0	0	0	7	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	17
eng453	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng454	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng455	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng456	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng457	0	0	0	0	0	0	0	0	0	0	29	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	30
eng458	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4
eng459	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4
eng460	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng461	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng462	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
eng463	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng464	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng465	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng466	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng467	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	2
eng468	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3
eng469	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
eng470	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
eng471	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	2
eng472	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	5	0	0	0	0	0	0	0	0	0	0	0	0	13
eng473	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1
eng474	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	2
eng475	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1
eng476	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1
eng477	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	2
eng478	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	2
eng479	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	0	0	0	0	0	0	0	0	0	7

eng480	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	2
eng481	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
eng482	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1
eng483	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1
eng484	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1
eng485	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
eng486	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	15	21
eng487	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
eng488	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
eng489	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	2
eng490	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	3
eng491	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2
eng492	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
eng493	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
eng494	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
eng495	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
eng496	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Max:	188	224	152	151	268	278	141	211	169	76	76	83	76	74	137	131	78	108	72	70	56	47	55	61	66	128	97	79	64	48	62	64	49	75	84	83	74	86	82	109	117	81	1703

B.4 Duration of Contribution & Average Loading

The table below is a summary of each individual's contribution period to the project as revealed by the database, including average number of change requests written per 1000 while active in the program. Several engineers have a zero average, and appear due to having been mentioned in or assigned change requests, but having never written one.

	First	Last	Duration	Avg		First	Last	Duration	Avg
eng001	1	3	3	8	eng049	2	5	4	9.75
eng002	1	2	2	3.5	eng050	1	42	42	1.167
eng003	1	1	1	57	eng051	4	21	18	0.833
eng004	1	1	1	5	eng052	4	40	37	16.32
eng005	2	40	39	9.103	eng053	4	27	24	14.04
eng006	1	39	39	3.718	eng054	1	12	12	113.5
eng007	1	3	3	4	eng055	2	42	41	27.63
eng008	1	36	36	21.94	eng056	6	14	9	1.889
eng009	1	41	41	2.244	eng057	2	42	41	23.1
eng010	1	41	41	18.46	eng058	3	41	39	11.28
eng011	1	3	3	6	eng059	1	1	1	3
eng012			1	0	eng060	4	36	33	1.667
eng013	1	39	39	22.54	eng061	4	38	35	0.629
eng014	5	42	38	1.184	eng062	1	37	37	9.486
eng015	1	40	40	28.03	eng063	2	5	4	2.5
eng016	1	42	42	16.98	eng064	1	28	28	9.036
eng017	1	42	42	18.05	eng065	1	42	42	5.714
eng018	2	12	11	17.18	eng066			1	0
eng019	1	42	42	30.74	eng067	11	12	2	2
eng020	1	41	41	0.829	eng068	5	17	13	8.923
eng021	1	42	42	19.33	eng069	7	23	17	6.529
eng022	1	42	42	8.452	eng070	3	38	36	5.556
eng023	1	9	9	20.22	eng071	3	40	38	5.447
eng024			1	0	eng072	6	39	34	3.176
eng025	1	1	1	1	eng073	4	40	37	4.081
eng026	1	1	1	2	eng074	6	35	30	2.067
eng027			1	0	eng075	1	42	42	13.14
eng028	1	39	39	11.69	eng076	1	6	6	7.333
eng029			1	0	eng077	1	42	42	0.714
eng030	1	39	39	14.92	eng078	3	42	40	25.58
eng031	1	38	38	4.368	eng079	1	42	42	24.12
eng032	1	30	30	22	eng080	5	6	2	13
eng033	1	42	42	2.143	eng081	4	42	39	28.21
eng034	1	40	40	24.7	eng082			1	0
eng035	1	6	6	1.5	eng083	1	8	8	7.75
eng036	1	17	17	35.59	eng084	32	32	1	1
eng037	1	42	42	2.048	eng085	2	42	41	9.439
eng038	1	1	1	28	eng086	4	42	39	23.54
eng039	1	2	2	8.5	eng087	1	42	42	16.43
eng040	1	15	15	14.8	eng088	1	41	41	6.073
eng041	2	27	26	11.38	eng089	1	6	6	20.83
eng042	1	41	41	4.439	eng090	4	33	30	0.533
eng043	4	42	39	17.51	eng091			1	0
eng044			1	0	eng092	2	4	3	7.333
eng045	5	25	21	7.762	eng093	5	20	16	0.125
eng046	1	42	42	16.31	eng094	1	4	4	54
eng047	4	19	16	0.125	eng095	2	35	34	3.059
eng048	1	40	40	32.98	eng096	2	24	23	2.217

	First	Last	Duration	Avg
eng097	2	40	39	0.897
eng098	8	34	27	13.07
eng099	1	39	39	2.872
eng100	4	42	39	9.333
eng101	3	39	37	0.459
eng102	6	20	15	3.6
eng103	1	41	41	2.073
eng104	1	27	27	10.04
eng105	1	42	42	6.452
eng106	1	39	39	7.051
eng107	1	8	8	0.625
eng108	1	8	8	13.88
eng109	6	18	13	4.615
eng110	6	38	33	2.424
eng111	1	1	1	23
eng112	1	1	1	12
eng113	1	30	30	2.567
eng114	1	3	3	11.33
eng115	1	1	1	1
eng116	1	1	1	2
eng117	1	25	25	13.84
eng118			1	0
eng119	1	1	1	1
eng120	1	3	3	5
eng121	1	41	41	34.93
eng122	1	42	42	29.12
eng123			1	0
eng124			1	0
eng125			1	0
eng126	2	13	12	0.75
eng127	5	25	21	5
eng128	1	1	1	4
eng129	1	12	12	15.33
eng130	1	4	4	11.5
eng131	1	2	2	6.5
eng132	2	4	3	17
eng133	2	2	1	16
eng134	2	7	6	7.167
eng135	2	21	20	3.3
eng136	3	27	25	0.6
eng137	2	25	24	2.917
eng138	6	29	24	3.167
eng139	2	25	24	2.167
eng140	2	39	38	30.55
eng141	1	2	2	4
eng142	2	17	16	12.31
eng143	2	5	4	1.25
eng144	5	13	9	1.222
eng145	5	11	7	9.429
eng146	2	10	9	0.556
eng147	4	5	2	2.5
eng148	4	31	28	30.18
eng149	1	1	1	8
eng150	1	1	1	5
eng151	1	1	1	4
eng152	1	3	3	2.667
eng153	1	2	2	17
eng154	1	9	9	1.889
eng155	2	3	2	9.5
eng156	2	2	1	7
eng157	2	2	1	1
eng158	2	2	1	13
eng159	2	37	36	2.639
eng160	2	39	38	3.211
eng161	3	3	1	4
eng162	3	9	7	11.43
eng163	3	3	1	3
eng164	3	25	23	1.348
eng165	3	4	2	2
eng166	4	4	1	8
eng167	4	21	18	3.944
eng168	2	31	30	52
eng169	3	42	40	6.175
eng170	4	34	31	0.065
eng171	4	4	1	8
eng172	4	42	39	19.82
eng173	4	13	10	5.6
eng174	2	29	28	18.89
eng175	4	13	10	7.2
eng176	4	42	39	7.769
eng177	5	42	38	17.16
eng178	4	39	36	20.36
eng179	5	5	1	1
eng180	2	31	30	1.533
eng181	1	36	36	7.5
eng182	5	5	1	12
eng183	2	42	41	41.54
eng184	5	12	8	0.375
eng185	5	31	27	0.111
eng186	4	32	29	6.586
eng187	5	25	21	18.29
eng188	5	39	35	4.429
eng189	4	36	33	6.303
eng190	5	5	1	1
eng191	2	6	5	2.4
eng192	4	20	17	12.35
eng193	5	6	2	3
eng194	6	14	9	1
eng195	2	42	41	12.15
eng196	2	6	5	0.6
eng197	6	25	20	9.9
eng198	4	24	21	9.143
eng199	6	6	1	6
eng200	6	42	37	13.05
eng201	4	42	39	26.72
eng202	5	15	11	16.09
eng203	3	7	5	21.8
eng204	2	31	30	2.267
eng205	7	40	34	0.529
eng206	7	34	28	10.04

	First	Last	Duration	Avg
eng207			1	0
eng208	5	15	11	7
eng209	7	23	17	8.059
eng210			1	0
eng211	5	42	38	3.711
eng212	7	42	36	20.25
eng213	8	41	34	4.706
eng214	8	21	14	0.5
eng215	8	18	11	10.27
eng216	3	39	37	17.46
eng217	8	8	1	6
eng218	8	25	18	1.389
eng219	6	33	28	5.929
eng220	5	23	19	7.842
eng221	8	27	20	0.15
eng222	4	40	37	14.35
eng223	8	39	32	2.594
eng224	8	11	4	6.25
eng225	8	29	22	4.318
eng226	5	33	29	1.483
eng227	4	40	37	11.41
eng228	4	42	39	3.538
eng229	9	42	34	4.059
eng230	8	21	14	4.714
eng231	5	42	38	10.61
eng232	5	40	36	22.78
eng233	9	9	1	4
eng234	4	38	35	6.257
eng235			1	0
eng236	9	42	34	18.97
eng237	9	9	1	1
eng238	9	26	18	2.111
eng239	7	14	8	5.25
eng240	9	19	11	0.636
eng241	9	12	4	5.75
eng242	6	9	4	3
eng243	9	10	2	2
eng244	8	42	35	9.314
eng245	8	13	6	6.333
eng246	10	42	33	2.121
eng247	7	29	23	12.96
eng248	5	39	35	10.6
eng249	10	16	7	2.286
eng250	8	40	33	6.727
eng251	8	36	29	10.97
eng252	10	17	8	3.125
eng253	2	41	40	4.775
eng254	11	40	30	8.467
eng255	11	11	1	2
eng256	11	11	1	2
eng257	11	22	12	2.25
eng258	7	36	30	9.3
eng259	11	39	29	8
eng260	9	42	34	14.88
eng261	7	40	34	7.118
eng262	11	19	9	3.333
eng263	8	42	35	9.743
eng264	9	39	31	17.35
eng265	11	24	14	2.071
eng266	11	24	14	4.786
eng267	9	12	4	4.75
eng268	11	22	12	0.5
eng269	6	42	37	4.027
eng270	6	26	21	2.143
eng271	3	25	23	7.913
eng272	12	21	10	0.3
eng273	12	23	12	0.333
eng274	12	12	1	1
eng275	4	42	39	18.95
eng276	11	32	22	11.36
eng277	11	15	5	5.2
eng278	9	40	32	12.34
eng279	15	16	2	2
eng280	11	13	3	4
eng281	13	42	30	1.2
eng282	11	36	26	11.69
eng283	8	31	24	6
eng284	13	13	1	2
eng285	4	15	12	2.167
eng286	12	42	31	8
eng287	14	39	26	10.81
eng288	14	14	1	1
eng289	5	32	28	0.929
eng290	12	39	28	4
eng291	12	39	28	7
eng292	9	36	28	0.286
eng293	14	15	2	2.5
eng294	14	38	25	9.48
eng295	14	20	7	5.571
eng296	6	39	34	8.029
eng297	14	20	7	7.286
eng298	3	39	37	0.486
eng299	15	42	28	19.71
eng300	11	23	13	9.308
eng301	14	42	29	18.93
eng302	15	42	28	23.46
eng303	15	40	26	2.462
eng304	6	21	16	1.875
eng305	15	18	4	6.5
eng306	4	36	33	7.182
eng307	5	39	35	6.457
eng308	14	28	15	9.133
eng309	15	38	24	21.21
eng310	11	40	30	14.13
eng311	15	42	28	17.86
eng312	16	42	27	12.56
eng313	10	26	17	4.412
eng314	16	21	6	2.833
eng315	13	41	29	8.552
eng316	7	42	36	13.06

	First	Last	Duration	Avg
eng317	15	39	25	11.08
eng318	16	41	26	7.077
eng319	16	21	6	4.667
eng320	15	42	28	14.14
eng321	16	42	27	24.67
eng322	16	38	23	4.783
eng323	15	42	28	2.179
eng324	15	40	26	3.615
eng325	17	22	6	20.83
eng326	17	17	1	1
eng327	17	28	12	6.5
eng328	8	42	35	1.543
eng329	17	39	23	13.57
eng330	16	42	27	11.15
eng331	6	41	36	0.75
eng332	17	42	26	7.5
eng333	15	40	26	14.46
eng334	18	21	4	1.25
eng335	15	40	26	12.23
eng336	18	32	15	1.733
eng337	9	35	27	0.407
eng338	15	31	17	0.118
eng339	4	40	37	1.73
eng340	6	40	35	4.914
eng341	20	37	18	5.167
eng342	17	40	24	0.542
eng343	21	34	14	1.071
eng344	21	33	13	5.615
eng345	8	35	28	0.321
eng346	20	35	16	1
eng347	20	28	9	21.89
eng348	21	35	15	1.267
eng349	21	37	17	0.176
eng350	9	23	15	4.6
eng351	22	41	20	1.15
eng352	22	28	7	0.429
eng353	22	34	13	0.385
eng354	22	24	3	5
eng355	22	39	18	9.389
eng356	11	28	18	3.778
eng357	22	42	21	5.429
eng358	10	40	31	9.419
eng359	23	23	1	3
eng360	23	24	2	2
eng361	23	39	17	2.353
eng362	17	23	7	3.143
eng363	23	23	1	1
eng364	24	42	19	1.632
eng365	14	24	11	0.182
eng366	24	24	1	1
eng367	19	40	22	1.818
eng368	20	42	23	7.522
eng369	24	38	15	2.533
eng370	24	40	17	0.471
eng371	18	39	22	3.318
eng372	5	35	31	2.258
eng373	21	41	21	8.286
eng374	23	33	11	0.273
eng375	25	25	1	1
eng376	24	37	14	11.07
eng377	25	25	1	2
eng378	22	38	17	3.471
eng379	25	42	18	11.22
eng380	1	1	1	1
eng381	2	2	1	1
eng382	2	4	3	2.667
eng383			1	0
eng384	3	4	2	12.5
eng385	4	10	7	0.571
eng386	5	12	8	0.375
eng387	6	11	6	5.5
eng388	14	42	29	6.483
eng389	13	17	5	1.2
eng390	19	41	23	26.35
eng391	20	41	22	11.55
eng392	22	40	19	3.579
eng393	26	32	7	2
eng394	25	39	15	3.4
eng395	26	42	17	0.529
eng396	4	41	38	0.763
eng397	27	35	9	0.667
eng398	23	34	12	0.417
eng399	15	33	19	5.947
eng400	24	33	10	1.5
eng401	27	42	16	29.81
eng402	29	29	1	3
eng403	29	30	2	2.5
eng404	29	42	14	1.357
eng405	29	40	12	3.083
eng406	22	30	9	3.444
eng407	30	33	4	2.5
eng408	4	6	3	1.333
eng409	19	19	1	2
eng410	12	26	15	1.2
eng411	22	22	1	1
eng412	23	26	4	0.5
eng413	9	24	16	0.5
eng414	25	42	18	4.778
eng415	27	42	16	0.375
eng416	28	36	9	10.11
eng417	26	28	3	0.667
eng418	18	33	16	0.563
eng419	21	31	11	0.455
eng420	32	32	1	1
eng421	31	39	9	5.889
eng422	31	40	10	2.4
eng423	34	34	1	1
eng424	34	34	1	1
eng425	35	37	3	1.333
eng426	35	36	2	1

	First	Last	Duration	Avg
eng427	35	35	1	1
eng428	37	39	3	14.67
eng429	38	41	4	1.5
eng430	41	42	2	7.5
eng431	41	41	1	1
eng432	42	42	1	1
eng433	42	42	1	11
eng434	41	42	2	7
eng435	42	42	1	4
eng436	42	42	1	1
eng437	1	1	1	2
eng438	1	1	1	13
eng439	1	1	1	10
eng440	1	1	1	15
eng441	2	2	1	1
eng442	2	2	1	1
eng443	2	2	1	1
eng444	2	3	2	7
eng445	3	3	1	7
eng446	3	3	1	1
eng447	3	3	1	1
eng448	3	3	1	10
eng449	4	7	4	0.5
eng450	4	13	10	0.5
eng451	5	33	29	0.069
eng452	5	6	2	8.5
eng453	5	5	1	1
eng454	5	5	1	1
eng455	10	10	1	1
eng456	11	11	1	1
eng457	11	25	15	2
eng458	11	18	8	0.5
eng459	11	16	6	0.667
eng460	13	13	1	1
eng461	14	14	1	1
eng462	15	15	1	2
eng463	15	15	1	1
eng464	6	6	1	1
eng465	9	9	1	1
eng466	17	17	1	1
eng467	17	40	24	0.083
eng468	18	21	4	0.75
eng469	19	19	1	1
eng470	21	27	7	0.286
eng471	23	40	18	0.111
eng472	29	30	2	6.5
eng473	31	31	1	1
eng474	32	37	6	0.333
eng475	32	32	1	1
eng476	32	32	1	1
eng477	33	33	1	2
eng478	33	37	5	0.4
eng479	33	33	1	7
eng480	35	37	3	0.667
eng481	35	35	1	1

	First	Last	Duration	Avg
eng482	37	37	1	1
eng483	38	38	1	1
eng484	38	38	1	1
eng485	41	41	1	1
eng486	41	42	2	10.5
eng487	41	41	1	1
eng488	41	41	1	1
eng489	41	42	2	1
eng490	42	42	1	3
eng491	42	42	1	2
eng492			1	0
eng493			1	0
eng494			1	0
eng495			1	0
eng496			1	0

3369-4