Project Ignirtoq / Cheesegod : Querying facts from Source Code Repositories

Abram Hindle

Department of Computer Science

University of Victoria

abez@uvic.ca

August 3, 2004

This Presentation

- What am I going to cover?
  - Source Code Repositories
  - CVS
  - MRs
  - Questions that are applicable to source code repositories.
  - Ignirtoq / Cheesegod
  - Summary

Source Code Repositories

- Products
  - CVS
  - Subversion
  - Clear Case
  - Source Safe
  - BitKeeper
- Functionality
  - Revisions
  - Branches
  - Concurrency
  - Configuration Management

CVS

- Why CVS?
  - Defacto Standard for Open Source projects.
  - Many mature Open Source Projects have open repositories to study.
  - Learn about Open Source Software Development processes.

Operations

- CVS Operations
  - commit
  - update
  - checkout
- softChange attempts to track CVS Commits by grouping revisions.

MRs

- What is an MR?
  - Modification Request
  - Programmer submits a modification of the source code to the repository.
  - For CVS - when a programmer commits changes.

Questions

- What questions do we aim to answer?
  - Is it true that Programmer B only modifies files that Programmer A had modified previously
  - Is there no change to any files exceeds a certain number of lines.
  - Is there no change to any files is less than a certain number of lines.
  - There exists changes to files of Zero lines?
  - Is it true that all MRs with 200 or more files add the same number of lines per file?

Questions

- Can we validate Lehman's Laws on a project?
  - 1. The law of continuing change.
    - Any software system used in the real-world necessarily must change or become less and less useful in that environment.
  - 2. The law of increasing complexity.
    - As time flows forwards entropy increases. That is, as a program evolves its structure will become more complex. Just as in physics this effect can, through great cost, be negated in the short term.
  - 3. The law of large program evolution.
    - Program evolution is a self-regulating process and measurements of system attributes such as size, time between releases, number of reported errors, etc., reveals statistically significant trends and invariances.
  - 4. The law of organizational stability.

    - Over the lifetime of a program, the rate of development of that program is approximately constant and independent of the resources devoted to system development.
- 5. The law of conservation of familiarity.
- Over the lifetime of a system, the incremental system change in each release is approximately constant.
- (Lehman, M., (1980), 'Programs, life cycles and the laws of software evolution', Proc. IEEE, 15 (3).)

softChange

- What is softChange ?
  - softChange is a collection of applications that work together in order to further study the software evolution of a project.
  - softChange elaborates on data provided from many sources to enable an accurate description of the evolution of a project.
  - softChange helps answer common questions maintainers, developers and administrators have about a project.

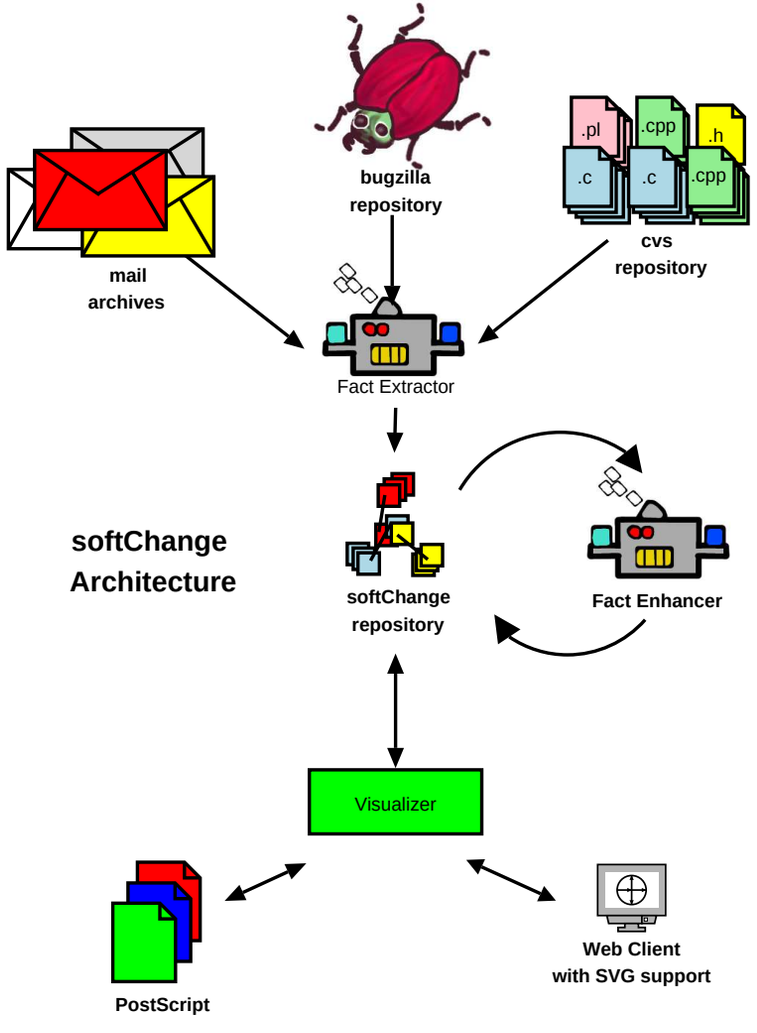

Figure 1: Architecture of Softchange

softChange

- What is softChange ?
  - Software Trails Repository - A relational database that stores all the historical data.
  - Software Trails Extractor - Extracts data from CVS, Changelogs, bug reports and emails.
  - Software Trails Analyzer / Fact Enhancer - Combines data in the repository to form MRs, and produce other useful statistics.
  - Visualizer - Visualize the data in the repository to aid the user in exploration and discovery.

Ignirtoq / Cheesegod

- What is Ignirtoq / Cheesegod
  - A model of source code repositories .
  - A query language to ask questions of the model
  - An engine which executes query on the model of a source code repositories .

Ignirtoq / Cheesegod

- Seriously what is with the name?
  - Ignirtoq - Inuit God of either:
    - Lightning (produced with a flint) - Lightning is appropriate as CVS branches over time and there is often concurrent and parallel development. (Non-wikipedia sources)
    - light and truth - the purpose of the system is allow you query source code repositories and get correct results. (From Wikipedia)
    - Culture still uses this god, it could be insensitive to use.
  - Cheesegod - A silly name
    - Silly
    - Cheesegod is some guy I know on the Internet, he works at a CVS Pharmacy.
    - cheesegod.com is a terrible site (bad design + quicktime embed)

- I have yet to decide on the actual name.

Ignirtoq / Cheesegod

- Model of MR Domain
  - Each MR is related to each other in time. There is a starting MR and an ending MR. Assume directed edges from one MR to the next MR in time. Assumption: No MR occurs at the same time as another MR.
  - MR’s have attributes id, log comment, date of revision, time of revision
  - MR’s are related to other elements from other domains. Authors, Revisions, Files

Ignirtoq / Cheesegod

- Model of Author Domain
  - Each Author is linked to MRs they authored.
    - Which are linked to revisions and files

Ignirtoq / Cheesegod

- Model of Revision Domain
  - Revisions the changes made to a file during from an MR
    - Linked to a MR
    - Linked to the file Changed
    - Contains changes data, number of lines change, revision id, revision number
    - Related to future revisions on the same files (branches and future revisions)

Ignirtoq / Cheesegod

- Model of File Domain
  - Revisions the changes made to a file during from an MR
    - Linked to other files in the same module
    - Linked to other files through revisions.

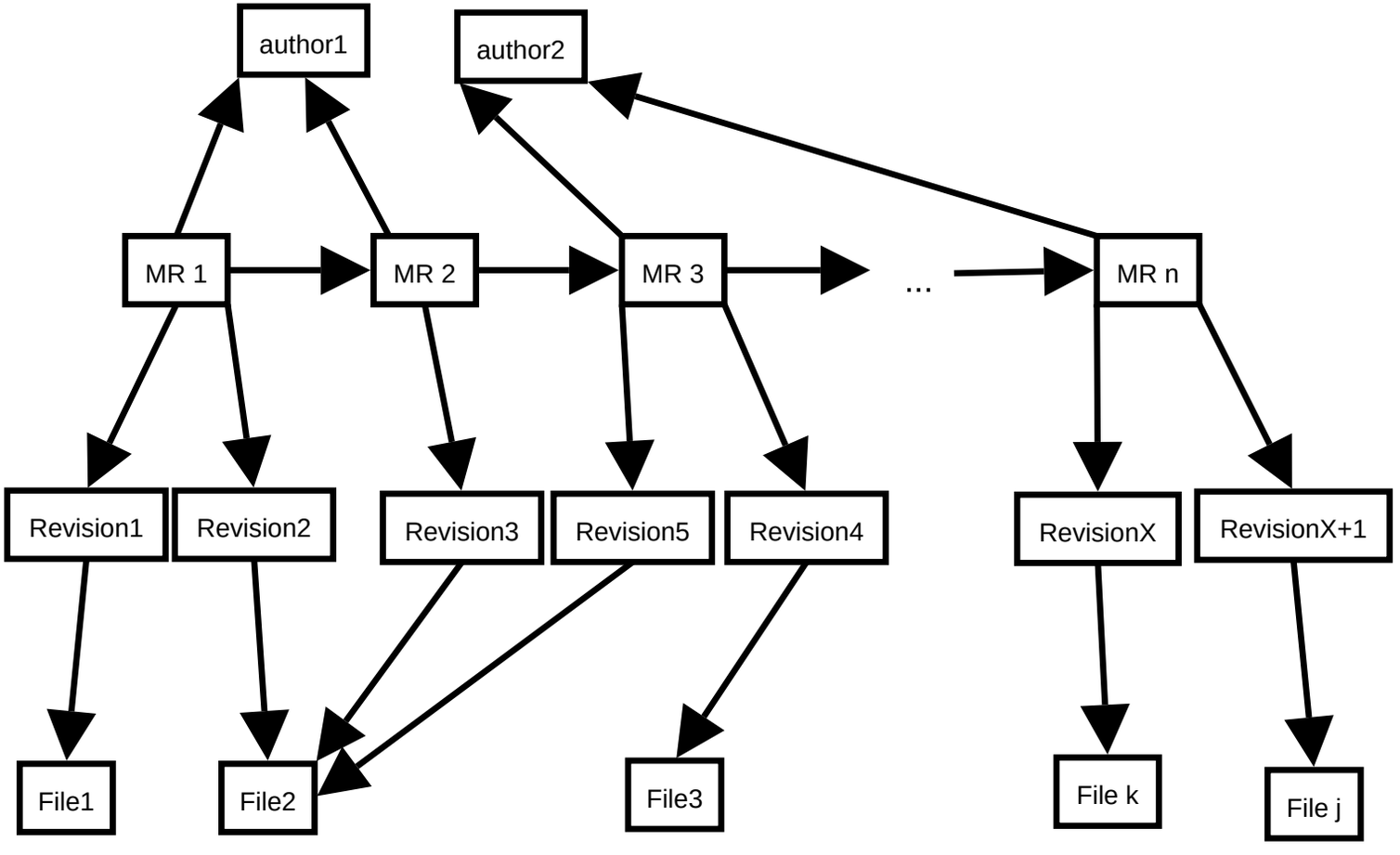

Figure 2: Model Graph Example

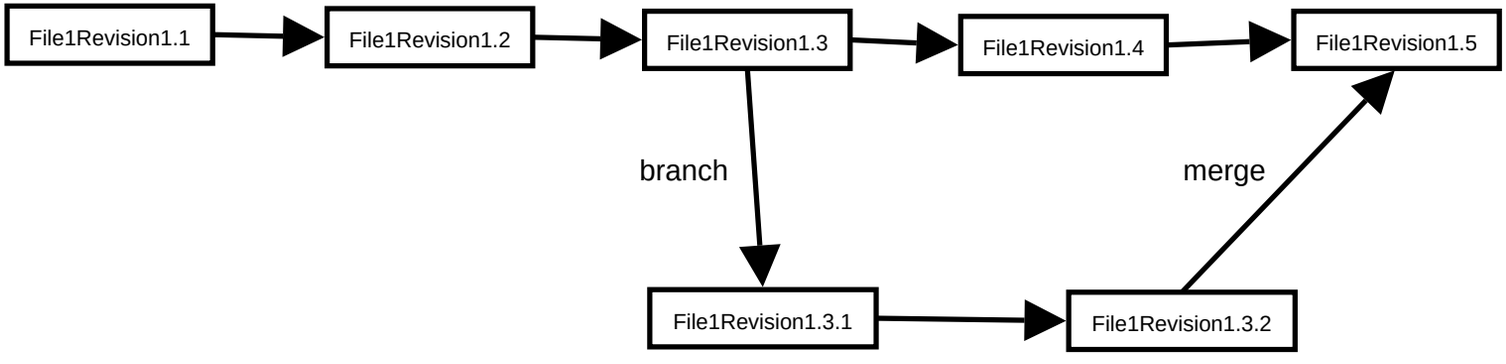

Figure 3: Revisions Graph Example

Ignirtoq / Cheesegod

- Query language
  - binary operators: $\rightarrow$, $==$, $\leq$, $\geq$, $>$, $<$, $!=$, $+$, $-$, $/$, $*$, $\&\&$, $||$
  - unary operators: $/$
  - Universal Quantifiers: $\forall_{before}$, $\forall_{after}$, $\forall$
  - Existential Quantifiers: $\exists_{before}$, $\exists_{after}$, $\exists$
  - Predefined Functions: $function(x)$
  - Properties: $x.property$
  - Identifiers : x

Ignirtoq / Cheesegod

- Query language examples
  - $\exists(a, MR)\{a\&\&a\}$
  - $\forall(b, MR)\{b.time > 0\}$
  - $\forall(b, MR)\{b.time <= 0\}$
  - $\forall(b, MR)\{b.time != 0\}$
  - $\forall(b, MR)\{b.time == 0\}$
  - $\exists(b, MR)\{b.totalMethodsBefore > b.totalMethodsAfter\}$
  - $\exists(b, MR)\{b.totalMethodsBefore != b.totalMethodsAfter||b.totalMethodsBefore != b.totalMethodsAfter\}$
  - $\exists(b, MR)\{(b.totalMethodsBefore > b.totalMethodsAfter)||(b.totalMethodsBefore <=$

$b.totalMethodsAfter)\}$

- $\forall_{before}(a, MR, a.time, a.time)\{a.totalMethodsAfter < a.totalMethodsAfter\}$
- $\forall_{after}(a, MR, a.time, 1000)\{a.totalMethodsAfter < a.totalMethodsAfter\}$
-

  $\exists(b, MR)\{\forall_{before}(a, MR, a.time, b.time)\{a.totalMethodsAfter < b.totalMethodsAfter\}\}$

Ignirtoq / Cheesegod

- Possible Uses:
  - Research, verifying Lehman’s laws, metrics
  - Fine Grained CVS Access Control
  - Invariant Testing
  - Invariant Discovery
  - Searching MRs (needs more set operators)
  - Legal Questions (e.g. SCO vs IBM)

Ignirtoq / Cheesegod

- Possible Use: Fine Grained CVS Access Control
  - One could specify invariants for the repository to limit the committing of code.
  - E.g. Users can only modify files they modified in the past, or files their teammates created.
  - Access control could depend on data in the repository rather than concrete individual rules about users.

Ignirtoq / Cheesegod

- Possible Use: Automated Invariant Discovery
  - Either randomly generate queries or iterate a subset of the query space searching for invariants.
  - Invariants could be tested across multiple projects. The most interesting invariants would be those which are true in most projects.
  - Can we get a better understanding about software evolution and the constraints on software evolution by learning about the invariants of a source code repositories ?
  - Huge search space.

Ignirtoq / Cheesegod

- Issues:
  - Sets and getting MRs out
    - Assume A and B are sets.. what does A + B mean?
  - Query Optimization ($O(n^2)$ queries and worse!)
  - Invariant Discovery takes a long time
  - Proof
    - Are all queries haltable?
    - If a query is answered in the model is it the same answer in the real system?

Future Work

- Better implementation
- Set support
- Actual Proofs
- Thesis

Summary

- Ignirtoq / Cheesegod will be used to ask questions of the source code repositories through a model of it.
- Needs more implementation and research
- To be truly useful we'll have to add set operations
- Many queries will take a lot of execution time
- Useful for:
  - access control
  - finding invariants

References