Detecting Interaction Coupling from Task Interaction Histories

Lijie Zou and Michael W. Godfrey
University of Waterloo

Ahmed E. Hassan
University of Victoria

Background

- Software maintenance is driven by *tasks*
  - Tasks have a *structure* (predefined / emergent)
  - Task structures implicitly include the set of program artifacts involved plus a rich set of relationships
- A *repository* of task structure info can be mined for latent knowledge about software development
  - But how to populate it? How to mine it?

Background

- Related work:
  - Logical coupling [Gall]
  - Interaction history [Parnin, Schneider, Robillard/Murphy]
  - Mylar [Kersten/Murphy]
  - NavTracks [Singer/Storey]

Recovering relations in task structure

- Murphy et al. [ECOOP-05]
  - *"A task structure consists of the parts of a software system and relationships between those parts that were changed to complete the task."*
- *Dependencies between changed artifacts* can be used to infer relations that are relevant to a task … but …
  - Change ⇏ Relevance
    - Some relations are unrelated to the task
  - Relevance ⇏ Change
    - Some patterns involved "just looking"

    e.g., viewing a function header before writing client code

Our approach: Interaction coupling

Definition

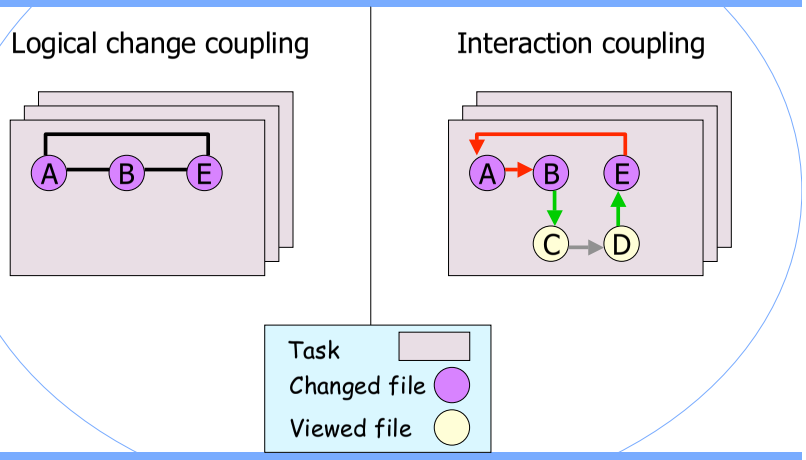
- Two artifacts that are "frequently" examined / changed "together" are likely to have some latent relationship.
  - We call this *interaction coupling*. *[cf. Gall et al.]*
- "frequently"
  - The weight of an IC is the total number of times that two artifacts are accessed together
  - We use a threshold value to throw out low valued couplings
- "together"
  - We track sequential accesses of entities, raised to the file level.
  - Together means "A then B" or "B then A" where each of A and B may be viewed or changed.

Interaction coupling

Intuition

- IC recovers relations within the context of a task execution
- Understanding of program structure leads to identifiable IC patterns
  - … which can improve our understanding of task structure
- Temporal windowing increases accuracy
- Weight reflects the significance and effort of a relation in a maintenance task

Interaction vs. change coupling



Interaction coupling patterns

Goal

- Use the info from ICs to improve understanding of maint. tasks, process, history, + status of the system

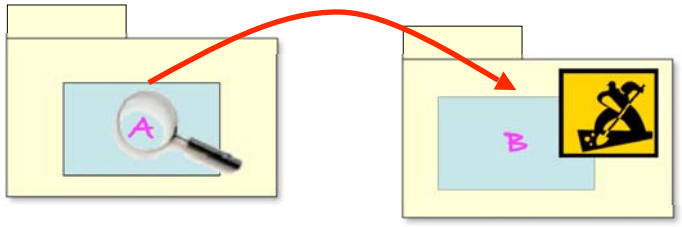
Idea

- IC patterns ⇒ maintenance task structure patterns

Process

- Recover interaction couplings
- Check whether an IC matches an existing pattern
  - Identify new patterns too

An IC pattern: Moving adaptation



Observable clues:

- B is a new file
- Methods in A are viewed-only when methods with identical names within B are changed
- A is eventually deleted

Case study

- A task interaction history repository built from our previous work
  - Same underlying data as reported in [WCRE-06]
- Three professional programmers working on two projects: *H* (577 classes, 57 KLOC) and *B* (202 classes, 20 KLOC)
- One month of data collection, plus follow-up interviews

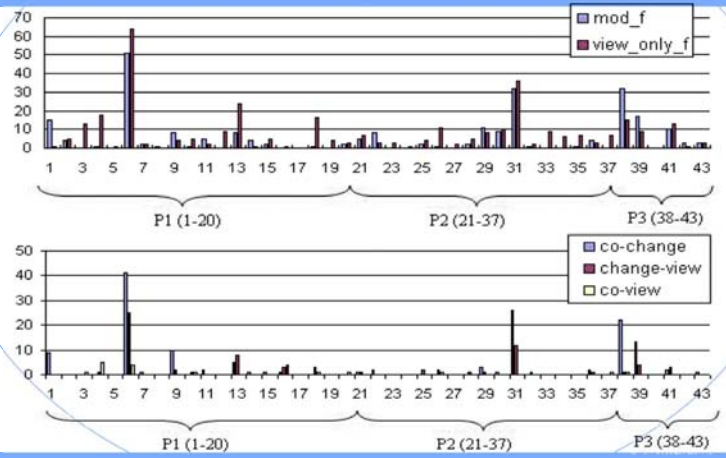
Research questions

Q1: Basic characteristics of interaction coupling

Q2: Interaction coupling patterns

Q3: Insights into the software system

Q1: Basic characteristics



Compare three IC types

Type	*# ICs*		*Weight*		
			Total	*Avg*	*Med*
Co-change	144	61%	1810	12.6	9
Change-view	72	30%	760	10.5	7
Co-view	20	9%	144	7.2	6
Total	236	100%	2714	11.5	8

Programmers concentrate on changes

Internal and external IC

Are two artifacts within the same subsystem?

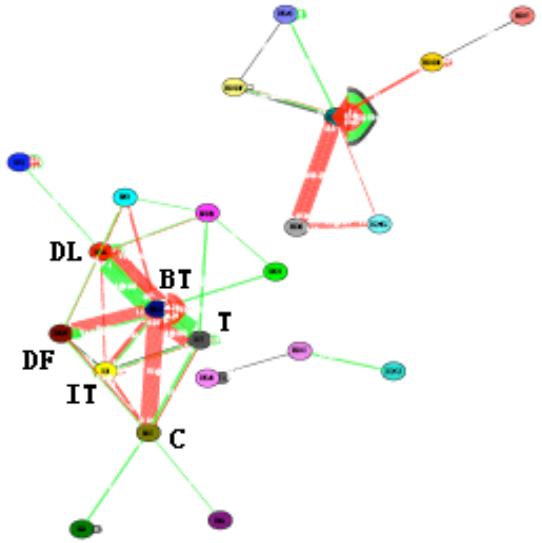
Type	*Count*			
	Total	*Co-change*	*Change-view*	*Co-view*
Internal	144	94	36	14
External	92	50	36	6

Large number of external links, to be analyzed

Q2: IC patterns

Focus on top 20% (46) strongest IC

Pattern	*#*	*Example (weight)*
Moving adaptation	4	BTManager – JBT (69)
Evolving interface	7	IDSession – JDSession (34)
Sibling cloning	3	JTRowSet – XTRowSet (16)
Superclass evolving	2	XServlet – DefaultXServlet (32)
Data management	3	TblCch – ETblCchManager (26)
Library referencing	2	JTRowSet – RowSet2XML (17)
Program test	3	JTRowSet – ADSessTests (37)
Peer concepts	2	Publication – Subscription (21)
Unrecognized	20	BTListener – DFEngine (57)



alysis

level

Case study: Summary

- IC patterns can aid in understanding *developer behaviour* during maintenance
- ICs can aid in understanding *emergent designs* in evolving systems

Concerns and threats to validity

- Is a time window of 2 events enough to be meaningful? [Parnin]
- Is "file-level" the right level of granularity?
- Professional developers, experts in their system
- One study, two small systems, three programmers, …
- Did the required instrumentation affect developer behavior?
  - Did the developers "keep the faith"? Were they "overly devout"?

Conclusions

- Interaction coupling (IC) is a useful concept in the modelling of task structure
  - Recovering relations in task structure
  - Towards a better understanding of task structure
- IC patterns help to improve our understanding of maintenance tasks
- IC can be used to recover latent knowledge about a software system

Detecting Interaction Coupling from Task Interaction Histories

Lijie Zou and Michael W. Godfrey
University of Waterloo

Ahmed E. Hassan
University of Victoria