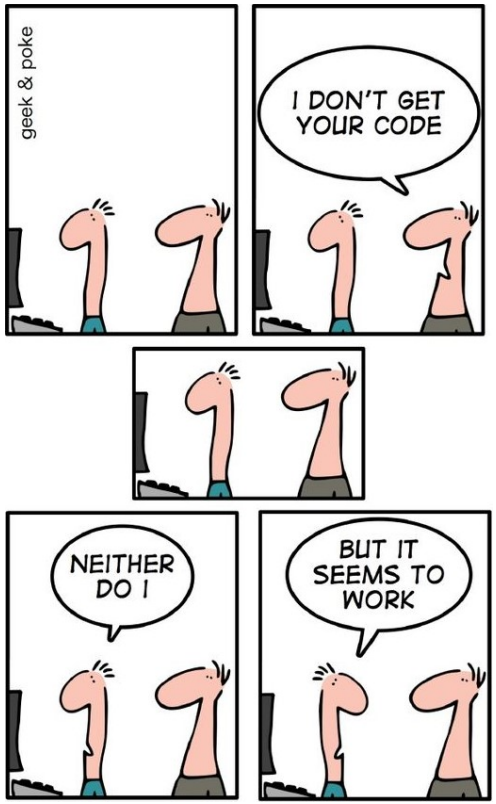


Herzlich Willkommen

Verteidigung der Bachelor Arbeit

Untersuchung von Fokus-Phasen in Paar-Programmierungs-Sitzungen

Freie Universität Berlin

Verteidigung Bachelor Arbeit

Untersuchung von Fokus-Phasen in Paar-Programmierungs-Sitzungen

Institut für Informatik FU Berlin

Arbeitsgruppe: Software Engineering

Gutachter: Prof. Lutz Prechelt

Zweitgutachter: Prof. Müller-Birn

Betreuer: Franz Zieris

Autor: Thomas Harms

Paar-Programmierung

In Paar-Programmierungs-Sitzungen produzieren zwei Programmierer gemeinsam ein Artifakt (Design, Algorithmus, Code). Die beiden Programmierer sind wie ein vereinter, intelligenter Organismus mit einem Verstand arbeitend, verantwortlich für jeden Aspekt des Artifaktes. [Der Schreibende] arbeitet [...] an der Tastatur und schreibt den Code. Der andere Partner observiert fortwährend und aktiv des Schreibendens Arbeit, erkennt Fehler, denkt über Alternativen nach, schlägt Ressourcen nach und strategische Ausrichtungen vor.

[R. R. Kessler, L. Williams, W. Cunningham, R. Jeffries., "Strengthening the case for pair programming". no. 4. s.l. : IEEE Sofware4, 2000. pp. 19-25. Vol. 17.]

- **zwei Programmierer** produzieren **gemeinsam** ein Artifakt
- **vereinter**, intelligenter Organismus **mit einem Verstand** arbeitend
- Unterscheidung in **„Driver“** und **„Observer“**

Problembeschreibung und Forschungsmethode

- Entwickler entwickeln Ideen gemeinsam
- Arbeitsteilung in „Driver“ und „Observer“ ist selten zu beobachten
- „Paar mit einem Verstand arbeitend“ ist ein ideales Ziel
- Aufbau und Aufrechterhalten von **Togetherness**

Forschungsfragen:

1. Warum treten so große Unterschiede bzgl. der Togetherness auf?
2. Welche Faktoren tragen dazu bei, das die Togetherness von Paaren hoch ist?

Forschungsmethode:

- Untersuchung von voraufgezeichneter PP-Sitzungen
- professionelle Entwickler deutscher Software-Unternehmen
- qualitative Datenanalyse auf der Basis „Grounded Theory Methodology“ (GTM)

Arbeitsflus
s

- Konzept eines der Resultate von Togetherness
- gut und leicht zu beobachten
- Beschreibt, wie schnell und flüssig Fortschritte im Arbeitsprozess erzielt werden

Drei Kategorien des Arbeitsflusses:

- PP_{normal} beschreibt einen erwarteten, normalen Arbeitsfluss. Es werden insgesamt Fortschritte erzielt, wobei sich die Geschwindigkeit volatil gestalten kann.

- PP_{fast} wird durch einen ungewöhnlich und extrem fließenden Arbeitsfluss mit sehr schnellem Fortschritt gekennzeichnet.

- $PP_{breakdown}$ zeichnet sich durch einen zähen Arbeitsprozess und/oder sehr uneffiziente
  Arbeitsweise aus.

Fokus-Phasen

- Abschnitte der Sitzung in PP_{fast}
- Togetherness wurde aufgebaut und wird während des gesamten Abschnittes auf einem hohem Niveau gehalten

Charakterisierung von Fokus-Phasen

- Bewertung der Aussagen hinsichtlich der Illuktionen
- Anwendung von Konzepten der Basis Schicht
- Kurze, schnelle Abfolge von Vorschlägen mit direkt folgender Zustimmung des Partners
- Ausbleiben von Ablehnung des Vorschlags oder Wissenstransfer

Rezeptartiges Abarbeiten

- Verabredung eines *„Rezeptes“* für den mindestens folgenden strategischen Abschnitt
- Rezept ist ein Schema, wie in den einzelnen taktischen Schritten verfahren wird
- keine Benennung über mehrere Arbeitsschritte: „Wie etwas geschehen soll“
- lediglich Erwähung des nächsten, zu betrachtenden Artifaktes
- Ergebnis hoher Togetherness und vorab verabredetem Rezept

- ermöglicht PP_{fast} und folgend Fokus-Phasen
- kann nur unter bestimmten Voraussetzungen von Fokus-Phasen auftreten

Vollständiges, detailliertes Verständnis des Problems

- Fokus-Phasen geschehen nicht zufällig oder willkürlich!
- Hohe Togetherness des Paares ist eine notwendige Bedingung für Fokus-Phase.
- Voraussetung für hohe Togetherness ist eine gemeinsame und bewußte Erarbeitung einer Lösungsstrategie.
- Gemeinsame Festlegung strategischer Ziele und deren Umsetzung ist vorab erforderlich.
- Beide Paartner müssen zwangsläufig ein vollständiges, detailliertes Verständnis der an die Sitzung gestellten Anforderungen und das Ziel haben.
- Offene Detailfragen haben oft Abhängigkeiten bzgl. weitere Teilaufgaben und führen zum Abbau von Togetherness und somit Rückgang zu PP_{normal} oder sogar $PP_{breakdown}$.

Spezifisches Wissen

- konzeptionalisiert das Verständnis von relevanten Aspekten des Software Systems
- Anforderungen, Architektur, Design, Design-Abhängigkeiten, bestehender Code und bekannte Fehler
- diverse beobachtete Konzepte setzen hohes spezifisches Wissen voraus
- bspw. führt hohes spezifisches Wissen zu effizienteren Phasen der Strategie-Findung
- Togetherness wird schneller und leichter aufgebaut
- kann während der Sitzung erworben werden

Bewußte Erarbeitung einer Lösungsstrategie

- Gemeinsame Erarbeitung einer Lösungsstrategie voraussetzend für PP_{fast}
- Falls dies „on the fly“ geschieht, ist volatile Togetherness zu beobachten
- bspw. rezeptartiges Abarbeiten setzt Erarbeitung eines Rezeptes voraus
- Togetherness wird während der Erarbeitung aufgebaut
- kann leichter aufrecht erhalten werden

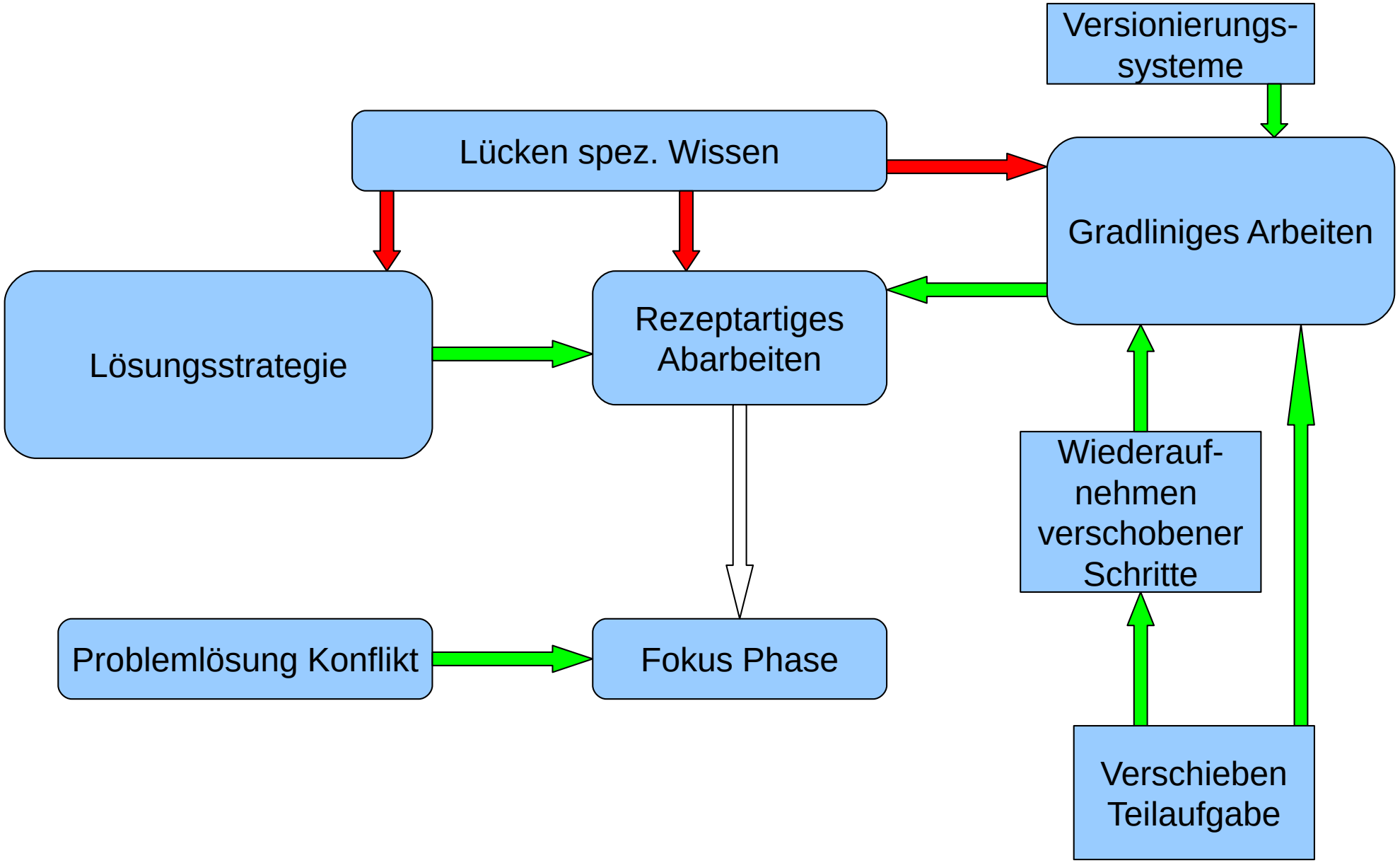
Gradliniges Arbeiten

- während einer Fokus-Phase muss Togetherness aufrechterhalten werden
- In Situationen von Unklarheiten, Uneinigkeiten usw. kann das akute Probleme verschoben werden.
- Togetherness bleibt somit hoch.
- Problem muss anschliessend wieder aufgenommen werden.
- profitiert von hoher Togetherness

Lösungsstrategien in Konfliktsituationen

- Konfliktsituationen treten erstaunlich oft auf
- insbesondere bzgl. Vorlieben oder Gewohnheiten von Prozessen oder Implementierungen, Algorithmen
- Togetherness geht stark zurück bis hin zu breakdown
- Togetherness muss anschliessend wieder aufgebaut werden

- Lösungsstrategie für effizienten Aufbau von Togetherness:
  - bewußtes Eintreten in Konfliktsituation
  - Vorschlag einer kurzfritsigen Lösungsstrategie
  - Vorschlag für einen Kompromiss einer mittlefristigen Lösung
  - in Aussicht stellen einer langfristigen Lösung, in Einbeziehung Dritter


Versionierungs-
systeme
Lücken spez. Wissen
Gradliniges Arbeiten
Lösungsstrategie
Rezeptartiges
Abarbeiten
Wiederauf-
nehmen
verschobener
Schritte
Problemlösung Konflikt
Fokus Phase
Verschieben
Teilaufgabe

Vielen Dank für Ihre Aufmerksamkeit!

Ich hoffe auf zahlreiche Fragen!

Fühlen Sie sich zu ausführlichen Diskussion eingeladen.

Herzlichen Dank an Franz Zieris für die Betreuung dieser Arbeit und zahlreiche Stunden der Diskussionen, ohne die diese Arbeit nicht möglich gewesen wäre.

Vielen Dank gebührt ebenfalls Professor Lutz Prechelt für die Begutachtung und Richtungsweisung, sowie Professor Claudia Müller-Birn.

product-oriented concepts

amend_design
Extend a given proposal regarding the structure and content of the program without rejecting the proposal.

ask_design
Ask for a concrete proposal regarding the structure and content of the program.

challenge_design
Reject a given proposal regarding the structure and content of the program and make an alternative proposal instead.

agree_design
Signal agreement with a given proposal regarding the structure and content of the program.

decide_design
Select one from among several alternative proposals regarding the structure and content of the program.

propose_design
Make one or several alternative proposals regarding the structure and content of the program.

disagree_design
Reject a given proposal regarding the structure and content of the program without making an alternative proposal.

remember_requirement
Remind the pair of a given (pre-specified) functional or non-functional requirement of the program.

challenge_requirement
Reject a given or proposed requirement and propose an alternative one instead.

agree_requirement
Signal agreement with a given or proposed requirement.

propose_requirement
Propose one or several alternative program characteristics that should be considered to be a requirement.

process-oriented concepts

amend_step
Extend a given proposal regarding the next tactical work step without rejecting the proposal.

ask_step
Ask for a concrete proposal regarding the next tactical work step.

explain_completion
Make a statement regarding the degree of completion of the current tactical work step.

challenge_step
Reject a given proposal regarding the next tactical work step and make an alternative proposal instead.

agree_step
Signal agreement with a given proposal regarding the next tactical work step.

agree_completion
Signal agreement with a statement regarding the degree of completion of the current tactical work step.

decide_step
Select one from among several alternative proposals regarding the next tactical work step.

propose_step
Make one or several alternative proposals regarding the next tactical work step.

challenge_completion
Reject a statement regarding the degree of completion of the current tactical work step and make an alternative statement.

disagree_step
Reject a given proposal regarding the next tactical work step without making an alternative proposal.

explain_state
Make a statement regarding the degree to which the current strategy or work plan has been worked through.

amend_strategy
Extend a proposed strategy or work plan without rejecting it.

ask_strategy
Ask for a concrete proposal regarding the strategy or work plan to be chosen.

agree_state
Signal agreement with a statement regarding the degree to which the current strategy or work plan has been worked through.

challenge_strategy
Reject a given proposal regarding the strategy or work plan and make an alternative proposal instead.

agree_strategy
Signal agreement with a given proposal regarding the strategy or work plan.

challenge_state
Reject a statement regarding the degree to which the current strategy or work plan has been worked through and make an alternative statement.

decide_strategy
Select one from among several alternative proposed strategies or work plans.

propose_strategy
Propose one or several alternative strategies or work plans.

propose_todo
Suggest that a certain work item will need to be taken care of later in the process.

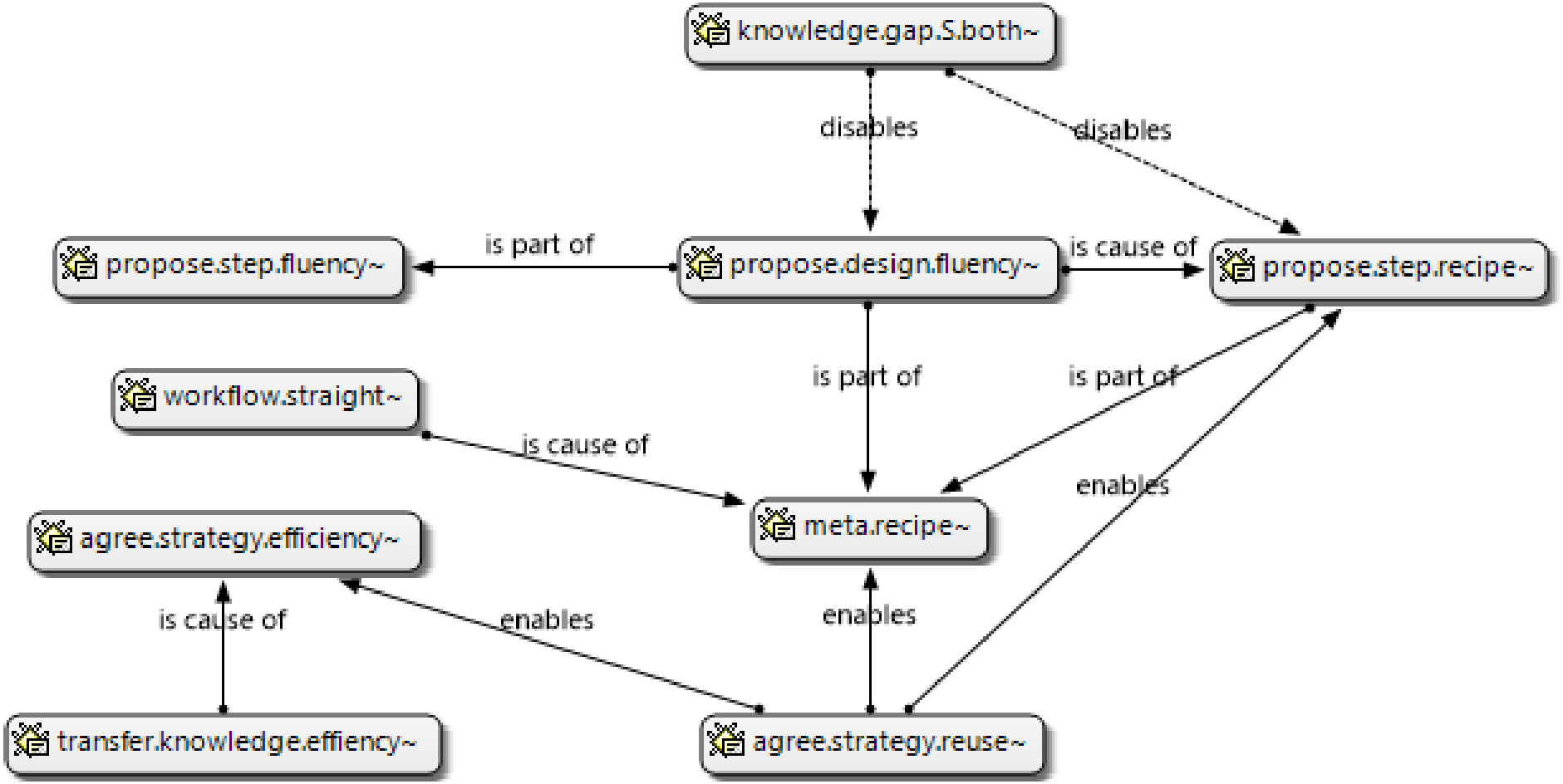
disagree_strategy
Reject a given proposal regarding the strategy or work plan without making an alternative proposal.

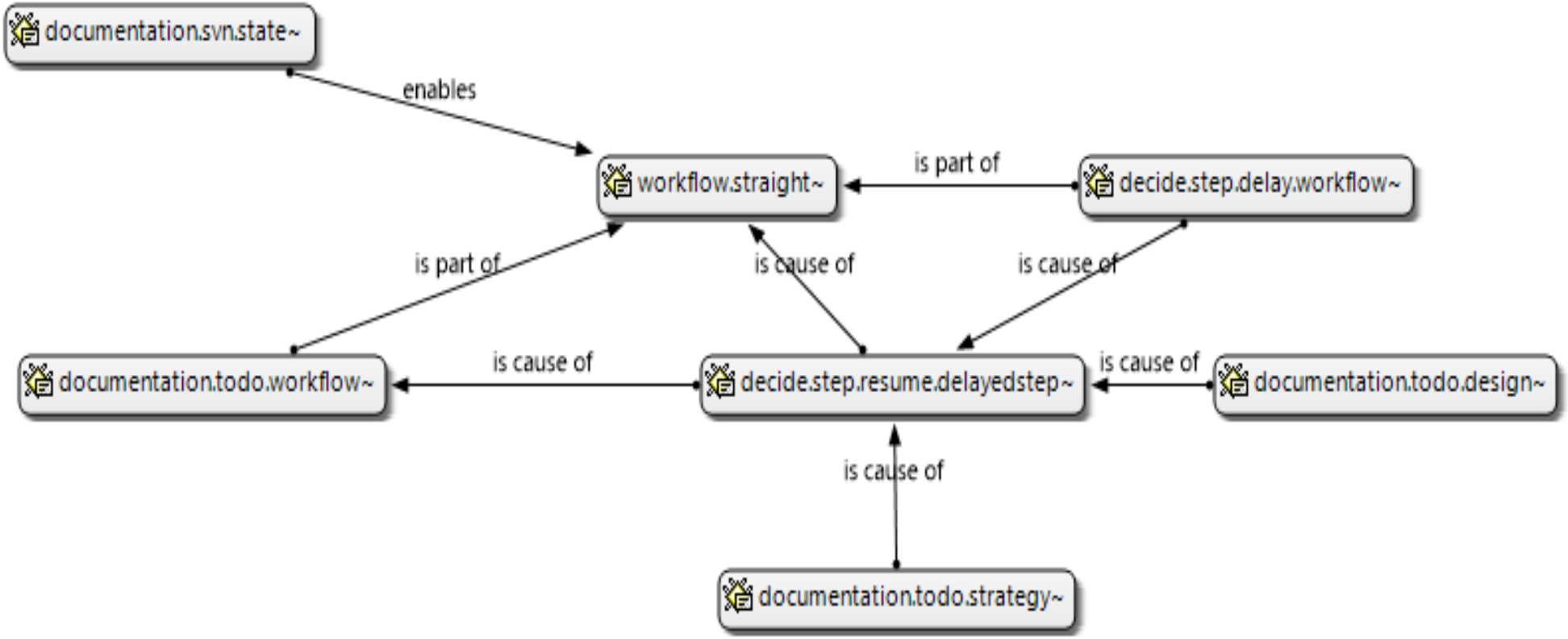
agree_todo
Signal agreement with a statement saying that a certain work item will need to be taken care of later in the process.

miscellaneous

mumble_sth
Make an incomprehensible utterance (highly fragmentary or acustically unclear).

say_off topic
Make an utterance that has nothing to do with solving the programming task.


knowledge.gap.S.both~
disables
disables
propose.step.fluency~
is part of
propose.design.fluency~
is cause of
propose.step.recipe~
is part of
is part of
workflow.straight~
is cause of
meta.recipe~
enables
agree.strategy.efficiency~
is cause of
enables
enables
transfer.knowledge.effiency~
agree.strategy.reuse~


documentation.svn.state~
enables
workflow.straight~
is part of
decide.step.delay.workflow~
is part of
is cause of
is cause of
documentation.todo.workflow~
is cause of
decide.step.resume.delayedstep~
is cause of
documentation.todo.design~
is cause of
documentation.todo.strategy~