Sketches and Diagrams in Practice

Sebastian Baltes and Stephan Diehl
University of Trier, Germany

@s_baltes
s.baltes@uni-trier.de
Sketches and Diagrams

Informal

Sketch

Diagram

Formal
Introduction

Past studies: Sketches and diagrams important in daily work of software developers

**Purpose:** Understanding, designing, communicating

[Cherubini07]

**Depict mental model of software**

[LaToza06]

**Medium:** Whiteboard, paper, computer

[Cherubini07, Walny11]

**Psychology:** Sketching augments information processing, sketches are sources of creativity

[Goldschmidt03, Tversky03]

Teams *improvise* representations, sketches/diagrams often *informal*

[Dekel07, Petre13]
Our Goal

Existing studies:
• Concentrated on certain aspects
• Single companies
• Academic environment
• Some had small number of participants

Our goal: Thorough description of how sketches and diagrams are used in software engineering practice

Better tool support for integrating sketches and diagrams into software development process
Our Goal

Existing studies:

- Concentrate on certain aspects
- Single companies
- Academia
- Some had small number of participants

Our goal: Thorough description of how sketches and diagrams are used in software engineering practice

Better tool support for integrating sketches and diagrams into software development process
Research Design

How to describe sketches and diagrams in SE practice?
Research Design

Existing Studies
- purpose, media, formality, etc.

Exploratory Field Study
- 3 companies
- 13 developers
- 47 sketches

11 Dimensions
- Revision
- Lifespan
- Effort
- Purpose
- Contributors
- Context
- Relation to Source Code
- Archiving
- Medium
- Formality
- UML-Elements

Online Survey
Online Survey

- **Target population:** "software practitioners"

- **Concise:**
  - ~10 minutes to complete
  - 28 questions, 15 about last sketch

- **Recruiting:**
  - Network of colleagues and contacts
  - Social networks
  - IRC channels and online communities
  - Directly contacted software companies
  - Article on major German IT news website  heise online
Participants

• n=394

• 32 countries
  • 54% Germany 🇩🇪 15% North America 🇺🇸 🇨🇦

• 52% software developers, 22% software architects

• Time spent developing software: 80% (median) ⏰

• Professional work experience: 10 years (median) 📊

• Software projects from various application areas
Results
Creation and Usage

When did you create your last sketch/diagram?

When did you use the last sketch/diagram created by some else?

Takeaway 1:
Creating own sketches and using sketches created by others are frequent tasks among software practitioners.
What medium did you use to create the sketch/diagram?

- Whiteboard (40%)
- Paper (18%)
- Computer (39%)
- Tablet (0.8%)
- E-Whiteboard (1.5%)

Analog (58%)

Digital (42%)
The sketch/diagram helped me to...
(multiple answers possible)

...design a new architecture (52%)
...design new features (48%)
...explain an issue to someone else (46%)
...analyze requirements (45%)
...understand an issue (44%)
The sketch/diagram helped me to...
(multiple answers possible)

...design a new architecture (52%)
...design new features (48%)
...explain an issue to someone else (46%)
**...analyze requirements (45%)**
...understand an issue (44%)
Effort and Revision

How much effective work time went into the creation and revision of the sketch/diagram up to now?

Revision:
15% revised once, 74% multiple times

Takeaway 2:
Most sketches are created in less than one hour and are revised at least once.
Please try to estimate the lifespan of the sketch/diagram (how long did/will you use it)?

Takeaway 3:
Almost half of the sketches are used for **at least several weeks**.
Formality and UML

**Formality:** Please try to specify the formality of your sketch/diagram.
(6-point Likert scale (0-5) from "very informal" to "very formal")

**UML:** To which degree does the sketch/diagram contain UML elements?
(6-point Likert scale (0-5) from "no UML elements" to "only UML elements")

68% informal
40% no UML
48% some UML
9% solely UML

Takeaway 4:
The majority of sketches and diagrams are informal. If UML is used, it is often mixed with other notations.
Please select the software artifact(s) to which the content of the sketch/diagram is related?
(multiple answers or no answer possible)

Most specific artifact:

- Methods: 23%
- Packages: 17%
- Projects: 19%
- Statements: 8%
- Attributes: 9%
- Classes: 20%

Level of abstraction:
- Statements (0%)
- Attributes (5%)
- Methods (10%)
- Classes (15%)
- Packages (20%)
- Projects (25%)
Please select the software artifact(s) to which the content of the sketch/diagram is related?
(multiple answers or no answer possible)
Relation to Source Code

**Help self:** Do you think that the sketch/diagram could help you in the future to understand the related source code artifact(s)?

**Help others:** ... help other software developers ...

(6-point Likert scale (0-5) from "It will definitely not help " to "It will definitely help")

![Bar Chart]

**Takeaway 5:**
About **half of the sketches are rated as helpful** to understand the related source code artifact(s) in the future.
Archiving

Three questions:

1. Has the sketch/diagram been archived or will it be archived?

   58% archived

2. If the sketch has been archived or will be archived, why do you want to keep it?

3. If the sketch has not been archived and won't be archived, why do you not want to keep it?

   • Answers analyzed using open coding
   • Extracted four categories for the answers to each question
   • One category for archiving practice
Archiving

Whiteboard (40%)
Paper (18%)
Computer (39%)
E-Whiteboard (1.5%)
Tablet (0.8%)

Takeaway 6:
Most digital sketches, but also more than one third of the analog sketches, are archived.
Archiving – Why?

If the sketch has been archived or will be archived, why do you want to keep it?

- Documentation
- Understanding
- Future Use
- Visualization

"It will be difficult to understand the code without the diagram."

"[The code] can be quickly understood due to the visual representation without hours of digging through complex source code."

"[The sketch] shows concepts that are not directly visible from code."
Archiving – Why?

If the sketch has been archived or will be archived, why do you want to keep it?

- Documentation
- Understanding
- Future Use
- Visualization

**Takeaway 7:**
Sketches are kept, because they document software, visualize it, and support its understanding.
Archiving – Why not?

If the sketch has not been archived and won't be archived, why do you not want to keep it?

- Served its purpose
- Substituted
- Outdated
- Technical Issue

"I do want to keep the sketch, but I have no way to archive whiteboard drawings."

"In case there was an easy way to combine both, code [...] and sketch I might have thought about archiving it."

"There is no good option to keep the sketch together with source code."
Archiving – How?

- MediaWiki
- git
- Subversion
- JIRA
- Bugzilla
- Email
Summary
Takeaways

1. Creating own sketches/diagrams **and using** sketches/diagrams created by others are frequent tasks among software practitioners.

2. Most sketches/diagrams are **created in less than one hour** and are **revised** at least once after creation.

3. Almost half of the sketches/diagrams are **used for at least several weeks**.

4. Majority of sketches/diagrams are **informal**.

5. About half of the sketches/diagrams are rated as **helpful** to understand the related source code artifact(s) in the future.

6. Most digital sketches/diagrams, but also more than one third of the analog ones, are **archived**.

7. Sketches/diagrams **document** the implementation, visualize it, and support its understanding.
Conclusion

- **Software documentation** is frequently **poorly written** and out of date
  [Forward02, Lethbridge03]

- Sketches and diagrams could serve as a **supplement** to conventional documentation

- Software practitioners are **willing to keep** their sketches and diagrams

- **Better tool support needed** for archiving and retrieving sketches/diagrams related to source code artifacts

- Tools should support **evolution** of sketches/diagrams (and software)
Future Work

1. What *helpful sketches* are helpful from not *helpful sketches*?

2. What *context information* is required to understand sketches later?

3. Do (informal) visualizations for certain source code artifacts share *common characteristics*?

**Recommendations** on how to create, augment, or annotate sketches so that they can serve as a valuable software documentation.
Questions?

Survey data and questionnaire available at:

http://st.uni-trier.de/survey-sketches

Reviewed and accepted by Artifact Evaluation Committee

@s_baltes

s.baltes@uni-trier.de