

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/275334482>

To Write a Conference Paper

Research · April 2015

DOI: 10.13140/RG.2.1.5078.4805

CITATIONS

0

READS

794

1 author:



Brian Berenbach

Georgia Institute of Technology

79 PUBLICATIONS 920 CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:



Modeling and Simulation in the Systems Engineering Life Cycle, Dr. Margaret Loper Editor, Springer 2015 [View project](#)



Unified Requirements Modeling Language (URML) [View project](#)



25th anniversary
annual INCOSE
international symposium
Seattle, WA
July 13 - 16, 2015



To Write a Conference Paper

Brian Berenbach
Georgia Tech Research Institute
brian.berenbach@gtri.gatech.edu



My Qualifications



- Over 35 peer reviewed conference papers
- Over 22 articles
- Reviewer for IEEE, ACM, and INCOSE conferences
- Track & workshop chair at several conferences
- 1 book + 1 invited book chapter
- “Best paper” awards
- First journal paper published May 1969

Why?



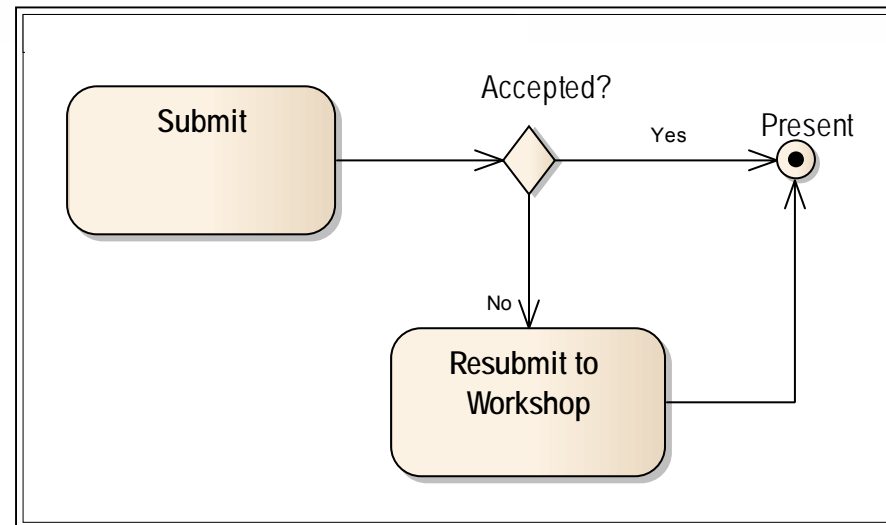
- Attend a conference
- Advance the art
 - Share Research
 - Share Experience
- Learn
- Network
- Advancement

WHY?

Strategy



- Acceptance Rate?
- Research or Practitioner Conference?
- Workshop or Main Track Paper?



Go it alone?



- One does not have to do the lonely-hero thing.
- Publishing with coauthors can be a plus
 - Extra pair of eyes to review the paper
 - More hours to put in->better quality
 - Help with formatting
- Downsides
 - the paper may not completely reflect *your* views
 - Coauthors may not meet their writing deadlines (Panic!!!)



Now that that is out of the way...

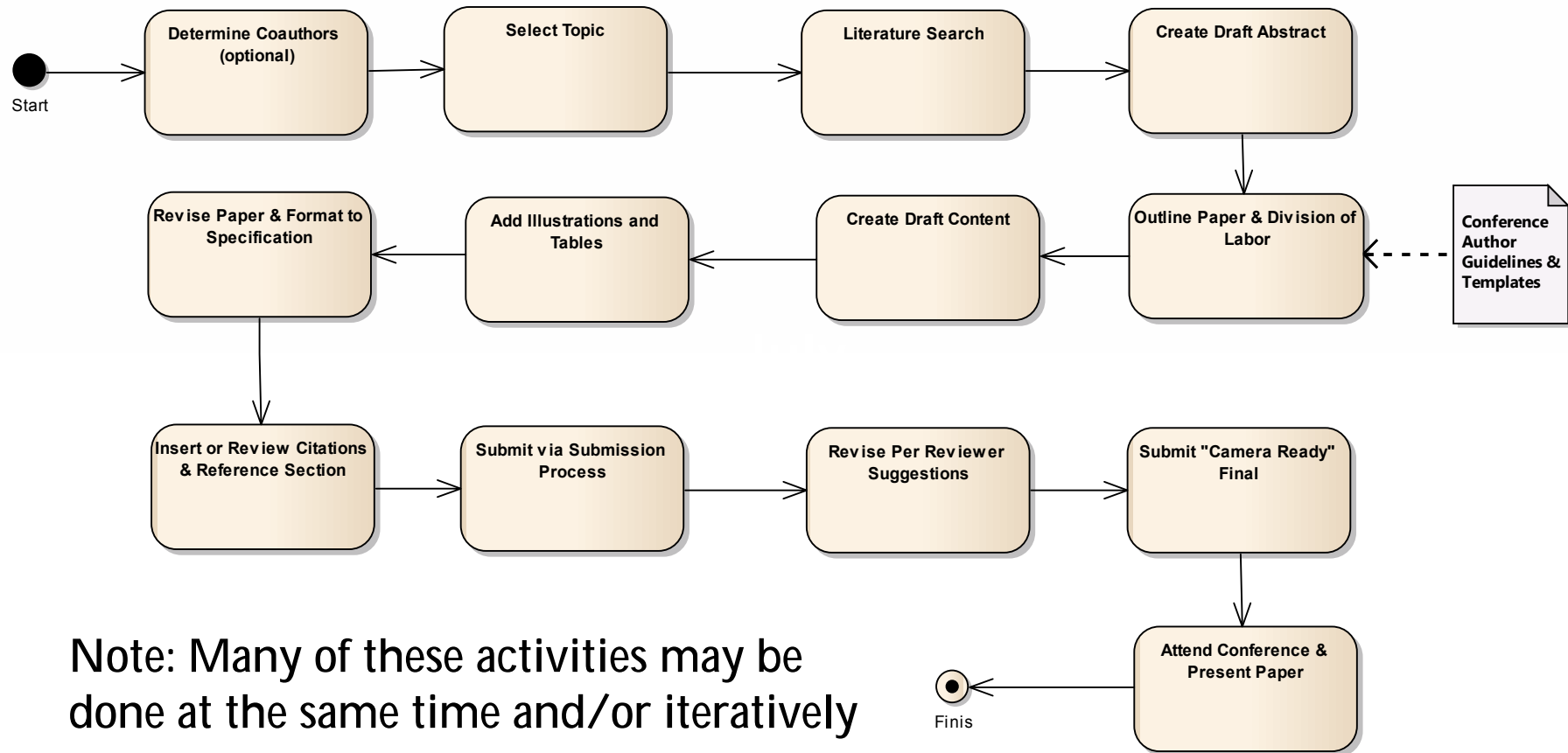


- Paper Writing Process
- Content
 - Abstract
 - Introduction
 - Body
 - Summary/results/conclusions
 - Acknowledgments
 - References
- Citations
- Formatting

Process- a “labor of love”



act [Package] Paper writing [Paper writing Process]



Note: Many of these activities may be done at the same time and/or iteratively

Select Topic



- The topic should reflect the actual research, discovery or practice.
- It should be novel or an improvement on existing art (i.e. why publish?)
- The subject should be of interest to the target audience.
- There should be some benefit to the community (i.e. positive results)

Literature Search



- The literature search needs to be comprehensive to avoid publishing something already reported, or slighting people who have made similar advances.
- When looking for citations, do not use indirection, but when coming across an interesting reference, go back to the original source of the reference for a citation.
- Be careful to only use original references. The Wikipedia is generally not considered a reliable source.
- Added bonus: sometimes a literature search can be expanded to a paper in its own right, e.g. a survey of the state-of-the-art.



Literature Search



Tabular Notations for State Machine-Based Specifications*

Markus Herrmannsdörfer, Dr. Sascha Konrad, and Brian Berenbach
Siemens Corporate Research

Finite state machines are a widely used concept for specifying the behavior of reactive systems. Numerous graphical notations based on finite state machines have been developed and are commonly used today, such as state transition diagrams, Harel statecharts, and Unified Modeling Language (UML) state machine diagrams. While not as widely used, tabular notations for state machine-based specifications offer complementary advantages to diagrammatic notations. In this article, we describe five approaches using tabular notations for state machine-based specifications and evaluate their approaches for use in software development.

The term *reactive system* describes a system that needs to continuously react to inputs coming from the environment. Finite state machines are a widely used concept for specifying the behavior of such systems. Since finite state machines allow the rigorous capture of functional aspects of system behavior, they offer several advantages over informal specifications. For example, they provide the ability to automatically generate code or test cases, and they enable formal verification and validation (V&V). Generally, a finite state machine is an appropriate representation when a problem or solution has the following characteristics:

- Finite and discrete set of states (e.g., on, off, and standby).
- Discrete and manageable set of inputs.
- Change of state is only performed in response to an input (e.g., if a button is pressed, then the machine transitions from state off to state on).

State machines are used for specifying functional properties for a wide variety of systems, such as control systems and user interfaces. For example, Siemens uses state machines to precisely specify the circuitry in mail sorting systems and the controls in car radios. They are also the paradigm of choice for software complex design and programmatic interpretation of natural language. Numerous graphical notations for state machines have been developed and are commonly used today, such as state transition diagrams, Harel statecharts [1], and UML state machine diagrams [2]. Graphical notations are often preferred by developers, analysts, and testers over textual information, since diagrams allow the visualization of complex relationships.

Tabular notations for state machines (commonly also referred to as *state tables* or *state transition tables*) offer complementary advantages to these graphical notations. For example, the incompleteness of a specification, i.e., the actions of the system in a specific state in response to a specific event that are not addressed by the specification, can easily be identified as empty cells in the table. In addition, tabular notations are relatively compact and have shown to scale well to practical systems [3]. Due to these reasons, tabular notations for state machines are preferred in some domains over graphical notations for the rigorous specification of system behavior. For instance, Siemens Automotive commonly receives system requirements in form of state tables, captured in either Excel sheets or proprietary databases.

While a tabular representation is relatively compact and the completeness of the requirements specification can easily be determined, it has been shown to cause numerous difficulties. For instance, the requirements specification for a system of realistic size is often quite large and of considerable complexity, consisting of numerous large tables. As a result, precisely understanding the required behavior solely through visual inspection is difficult. Moreover, requirements captured in simple Excel sheets are difficult to analyze for consistency and adherence to critical properties.

This article presents and evaluates several state machine-based tabular notations that can address some of the aforementioned problems. For instance, some notations enhance the understandability of the specification by offering a complementary graphical representation. In addition, hierarchical composition is used by several notations to keep the specification tractable and some provide tool support for V&V. The remainder of this article is organized as follows: the Background section provides an overview of finite state machines and Harel statecharts. The Tabular Notations for State Machines section describes five approaches using tabular notations for state machine-based specifications. We conclude by evaluating these notations for use in software development with respect to several factors.

Background

This section introduces finite state machines, including a common graphical and tabular notation, and briefly describes the advanced features of Harel statecharts.

Finite State Machine

The term *finite state machine* describes a class of computational models that consist of a finite set of states, a start state, a set of inputs (events), and a transition function that determines the next state of the finite state machine based on the current state and input [4]. The finite state machine starts computation in the start state; transitions between states are performed based on the transition function. Numerous variants of this basic type of state machine exist. For example, Moore machines extend finite state machines with outputs (actions) associated with states, while Mealy machines associate outputs with transitions [5]. For the remainder of this article, we use Mealy machines as the computational model. Finite state machines may be deterministic or non-deterministic. In deterministic finite state machines for a given input, one transition can be taken from the current state, at most. In non-deterministic finite state machines, however, one input may enable several transitions of which one is then taken.

A common way of representing finite state machines is the use of *state transition diagrams* (commonly also referred to as *state diagrams*). State transition diagrams are directed graphs in which states are depicted as nodes and transitions are represented by directed edges. Transitions are commonly labeled with the triggering events and actions, using the following general syntax: *trigger/action(s)*. Figure 1 contains a sample state transition diagram showing the simple behavior of a door: The door can be opened or closed. If the door is closed, then it can be locked or unlocked. The door can only be opened when it is

Sample of a literature search that was extended into a survey paper

Create Draft Abstract



- The abstract should catch the reader's attention. Often the abstract is published separately, and can be used by readers in a compendium to determine if they want to see the full paper or attend the presentation
- It should summarize the content of the paper without giving away the “punch-line”. For example: “the results of the study should be of interest to any Java programmer.”
- Never put something in the abstract that is not in the body of the paper. For example, if “unique insights into the state of the art” appears in the abstract, then the “unique insights” must appear in the body of the paper.
- Use abstracts from oft cited papers as examples.

Sample Abstract



“Technical Debt” is a term first used by Ward Cunningham in an experience report in 1992. The term refers to the accruing debt or downstream cost that happens when short term priorities trump long term lifecycle costs. The term, when introduced, was used in the context of the development of software systems. However, since 1992, the field of systems engineering has evolved, and it has been found that technical debt also applies to the development and construction of systems. This paper takes a contrary view; technical debt is discussed mostly in the context of bad practices; the author contends that the focus should be on system principles that preclude the introduction, either anticipated or unanticipated, of negative lifecycle impacts. A set of heuristics is presented that describes what should be done rather than what should not be done. From these heuristics, some emergent trends will be identified. Such trends may be leveraged to design systems with reduced long term lifecycle costs and, on occasion, unexpected benefits.”*

***B. Berenbach, “On Technical Credit”, Procedia Computer Science 28 (2014) 505 – 512**

Copyright ©2015 Brian Berenbach All Rights Reserved.

25th anniversary
annual INCOSE
international symposium
Seattle, WA
July 13 - 16, 2015

Sample Abstract



Background

“Technical Debt” is a term first used by Ward Cunningham in an experience report in 1992. The term refers to the accruing debt or downstream cost that happens when short term priorities trump long term lifecycle costs. The term, when introduced, was used in the context of the development of software systems. However, since 1992, the field of systems engineering has evolved, and it has been found that technical debt also applies to the development and construction of systems. This paper takes a contrary view; technical debt is discussed mostly in the context of bad practices; the author contends that the focus should be on system principles that preclude the introduction, either anticipated or unanticipated, of negative lifecycle impacts. A set of heuristics is presented that describes what should be done rather than what should not be done. From these heuristics, some emergent trends will be identified. Such trends may be leveraged to design systems with reduced long term lifecycle costs and, on occasion, unexpected benefits.”

***B. Berenbach, “On Technical Credit”, Procedia Computer Science 28 (2014) 505 – 512**

Copyright ©2015 Brian Berenbach All Rights Reserved.

25th anniversary
annual INCOSE
international symposium
Seattle, WA
July 13 - 16, 2015

Sample Abstract



“Technical Debt” is a term first used by Ward Cunningham in an experience report in 1992. The term refers to the accruing debt or downstream cost that happens when short term priorities trump long term lifecycle costs. The term, when introduced, was used in the context of the development of software systems. However, since 1992, the field of systems engineering has evolved, and it has been found that technical debt also applies to the development and construction of systems. This paper takes a contrary view; technical debt is discussed mostly in the context of bad practices; the author contends that the focus should be on system principles that preclude the introduction, either anticipated or unanticipated, of negative lifecycle impacts. A set of heuristics is presented that describes what should be done rather than what should not be done. From these heuristics, some emergent trends will be identified. Such trends may be leveraged to design systems with reduced long term lifecycle costs and, on occasion, unexpected benefits.”

Controversy or
interesting facts

Sample Abstract



“Technical Debt” is a term first used by Ward Cunningham in an experience report in 1992. The term refers to the accruing debt or downstream cost that happens when short term priorities trump long term lifecycle costs. The term, when introduced, was used in the context of the development of software systems. However, since 1992, the field of systems engineering has evolved, and it has been found that technical debt also applies to the development and construction of systems. This paper takes a contrary view; technical debt is discussed mostly in the context of bad practices; the author contends that the focus should be on system principles that preclude the introduction, either anticipated or unanticipated, of negative lifecycle impacts. A set of heuristics is presented that describes what should be done rather than what should not be done. From these heuristics, some emergent trends will be identified. Such trends may be leveraged to design systems with reduced long term lifecycle costs and, on occasion, unexpected benefits.”


Value proposition for the paper

Conference Guidelines



SUBMISSIONS

Paper Submission

 **FINAL Paper Preparation Template 2015**

 **Paper Evaluation Criteria 2015**

 **IP Release Form 2015**

 **Submission Instructions 2015**


 **Paper Template**

Conference Guidelines



SUBMISSIONS

Paper Submission

 **FINAL Paper Preparation Template 2015**

 **Paper Evaluation Criteria 2015**

 **IP Release Form 2015**

 **Submission Instructions 2015**


Important! What the reviewers will be looking for.


Conference Guidelines



SUBMISSIONS

Paper Submission

 **FINAL Paper Preparation Template 2015**

 **Paper Evaluation Criteria 2015**

 **IP Release Form 2015**

 **Submission Instructions 2015**


The release form is mandatory! The author(s) may need approval by a company or agency for publication, and this can take a significant amount of time. One approach that may work is to have an academic coauthor.

Conference Guidelines



SUBMISSIONS

Paper Submission

 **FINAL Paper Preparation Template 2015**

 **Paper Evaluation Criteria 2015**

 **IP Release Form 2015**

 **Submission Instructions 2015**

Don't forget to follow the instructions exactly, particularly about format, use of templates, and PAGE LIMITS. Incorrectness with any of these can be grounds for summary rejection. Reviewer time is valuable, so any way to save a reviewer is pounced on.

Outline Paper & Divide Work



- Abstract
- Introduction
 - Background
 - Rationale
 - Summary of the paper
- What I/we did and why I/we did it
 - Process followed
 - Experiments (if any)
 - Threats to validity
- Results
 - What I/we found
 - What went well
 - What did not go well and why
- Summary & Conclusions
 - Summary of paper
 - Highlight of results and conclusions
- References

Outline Paper & Divide Work



- Abstract
- Introduction
 - Background/History
 - Rationale
 - Summary of the paper
- What I/we did and why I/we did it
 - Process followed
 - Experiments (if any)
 - Threats to validity
- Results
 - What I/we found
 - What went well
 - What did not go well and why
- Summary & Conclusions
 - Summary of paper
 - Highlight of results and conclusions
- References

Use results of literature
search here

Outline Paper & Divide Work



- Abstract
- Introduction
 - Background
 - Rationale
 - Summary of the paper
- What I/we did and why I/we did it
 - Process followed
 - Experiments (if any)
 - Threats to validity
- Results
 - What I/we found
 - What went well
 - What did not go well and why
- Summary & Conclusions
 - Summary of paper
 - Highlight of results and conclusions
- References

Other types of
papers might be
structured
differently

Outline Paper & Divide Work



- Abstract
- Introduction
 - Background
 - Rationale
 - Summary of the paper
- What I/we did and why I/we did it
 - Process followed
 - Experiments (if any)
 - Threats to validity
- Results
 - What I/we found
 - What went well
 - What did not go well and why
- Summary & Conclusions
 - Summary of paper
 - Highlight of results and conclusions
- References

When in doubt,
find oft cited
similar types of
papers and use a
similar structure

Introduction



- The introduction provides an overview for the history and current state of the art.
- The motivation for the work is described, including the value proposition for the paper.
- One or two paragraphs are used to provide an overview of the paper, e.g. what is written in each section
- Be mindful of the paper length, it is almost always capped. If the paper, after completion, is too long the introduction is one place to look to trim.

Create Draft Content



- A refereed paper is not an editorial.
 - No Assertion without Citation
 - Make opinion clearly visible.

Wrong: “Researchers have advocated for flexibility in design...”

Right: “Researchers such as deNeufville [16] have advocated for flexibility in design...”

Wrong: “An agile approach is better for this type of project”

Right: “In the authors’ view, an agile approach is better for this type of project...”
- Nothing will turn off a reviewer more than poor grammar and spelling.
 - It is a sign of sloppy technique and lack of attention to detail.

Summary & Conclusions



- Summarize the paper, i.e. “Paragraph One reviews the state of the art... Paragraph two compares ways of..., Section 3 describes how we found a cure for cancer...”
- Make sure the key points are called out and highlighted, not buried in a text discussion.
- Leave no room for doubt

Wrong:

blahblahblawediscoveredantigravityblahblahblawcureforcancerblahblahblah...

Right:

Our major findings:

- ☐ A cure for cancer
- ☐ An antigravity device
- ☐ ...

Add Illustrations & Tables



- Each illustration and table should support or illustrate key elements of the paper and not vice versa
- Each illustration and table should be cited and explained in the paper.
- Make sure that the illustrations, including lines and small text are visible both on the screen, and when the published paper is copied and printed in black and white

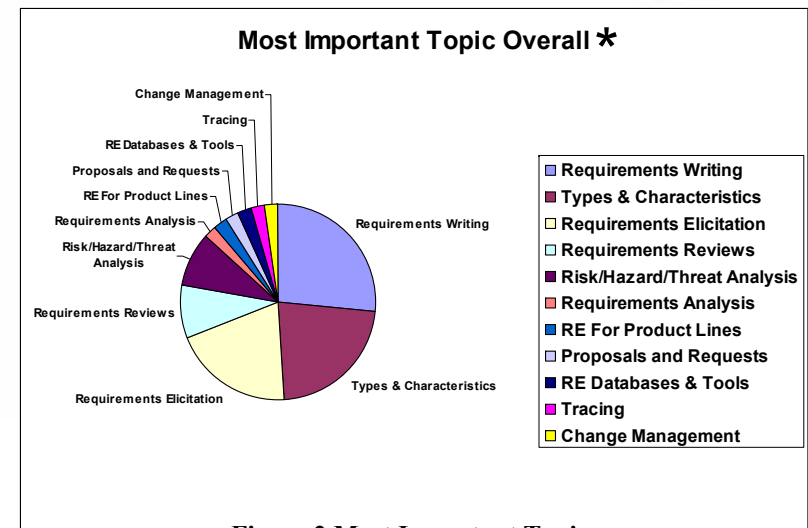


Figure 2 Most Important Topics

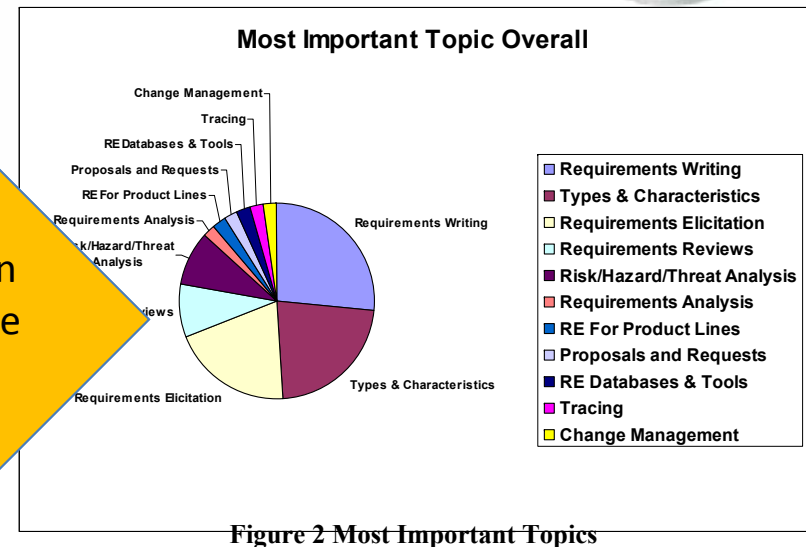
*Berenbach, B. and Rayment, T. "The Evaluation of a Requirements Engineering Training Program at Siemens", proceedings of the 16th IEEE International Requirements Engineering Conference (RE08), September 2008.

Add Illustrations & Tables



- Each illustration and table should support or illustrate key elements of the paper and not vice versa
- Each illustration should be cited and labeled in the paper.
- Make sure that the illustrations, including lines and small text are visible both on the screen, and when the published paper is copied and printed in black and white

Illustrations should not rely on color alone. Note that each pie section is labeled.



Add Illustrations & Tables



- Make sure that the illustrations, including lines and small text are visible both on the screen, and when the publication is printed in black and white.
- Each illustration and table should support or illustrate key elements of the paper and not vice versa
- Each illustration and table should be cited and explained in the paper.

Text should be large enough to see clearly in the paper when reading.

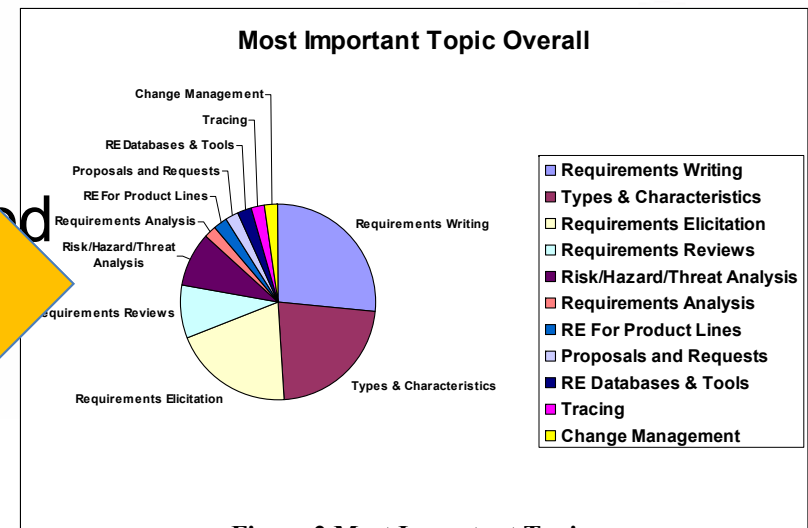


Figure 2 Most Important Topics

Add Illustrations & Tables



- Make sure that the illustrations, including lines and small text are visible both on the screen, and when the published paper is copied and printed in black and white
- Each illustration and table should support or illustrate key elements of the paper and not vice versa
- Each illustration and table be cited and explained in the paper.

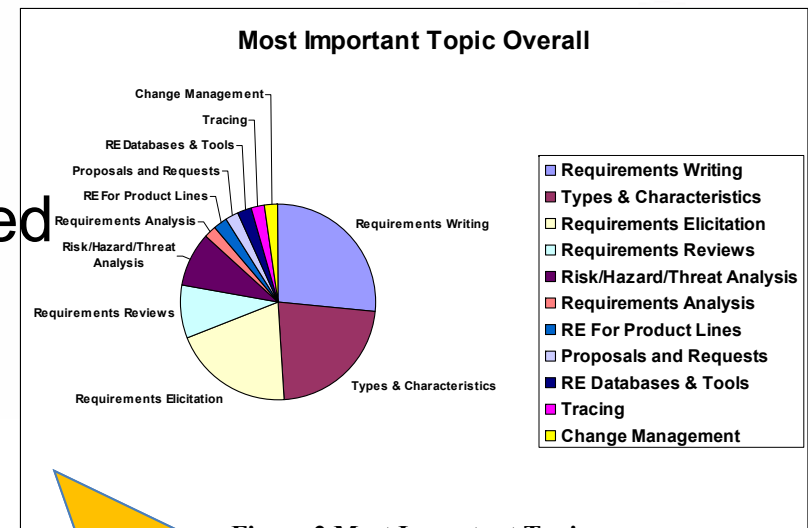


Figure 2 Most Important Topics

Illustration supports paper
and not vice versa



Revise Paper & Format (per template)

- If you are the sole author be sure to have colleagues review and comment
- If the conference language is not your first language, be sure to have a colleague whose native language is the conference language review carefully for spelling, word choice, idiomatic expressions and context.
- Be sure to include in an acknowledgement section, anyone who donated significant time to reviewing the paper or otherwise assisting. Note that anyone who contributed content should be an author.

Acknowledgement

The author gratefully acknowledges the assistance of Professor Dan Berry of the University of Waterloo for his comments and review of this paper.

Insert/Review Citations



- Every assertion not clearly the authors opinion must be cited.
- Citations should be in numeric order, e.g.
blahblahblah [1] blahblahblah [2] blahblah [3]...(unless the rules require a different order, e.g. alphabetical)
- The paper needs to be carefully reviewed to ensure that every assertion is cited, and that every reference is cited in the paper.
- Some formatters, e.g. LaTeX, give assistance and automate some of this work.

Create Reference Section



- Follow the conference rules for formatting references
- The references should be listed in the order cited
- Do not use the Wikipedia or other open changeable media for references (but if you have to, replicate the web page on your own web site)
- Take care with the web as web pages are not permanent, unlike printed text they can change with time.
- Use the original source in a reference. If, for an example you find an article with a quotation from another source and you want to use it, use the original source as your reference, and READ that source!

Submit “First Round”



- You will normally have to run your article through a “pdf checker”.
- Before you do that, have multiple reviewers read the paper, one extra pair of eyes is usually not good enough.
- Go for reviewers who are critical, preferably have published themselves.
- Submit via the conference mechanism, and be mindful of the deadlines, as they are usually *strictly enforced*.
- In an emergency, notify the appropriate conference chair and request permission for a late submission. Do this well in advance of the deadline.

Submit “Camera Ready”



- Take conference reviewer feedback to heart. Incorporate their feedback and suggestions.
- If your paper is not accepted, consider revising and resubmitting to a conference workshop (if offered)
- If that is not feasible, repair and resubmit either to a different conference, or the following year.
- Remember, not all conferences have the same acceptance rates!
- If you do submit elsewhere or again, make sure that you have modified the papers where possible as the reviewers suggest. Submitting an unchanged rejected paper is grounds for summary rejection.

Attend & Present



- You have to register to present
- You have to attend to present
 - Be mindful of the schedules
- When creating a presentation from the paper, make sure the illustrations will be visible to people in the back of the auditorium.
- Don't try to present the entire paper. Present enough to convince the audience to read the paper.
- Consider volunteering to be a session chair.



Potentially resubmit/expand

- If at first you don't succeed...



- But DO make the useful corrections of the reviewers before you resubmit!

Questions?



