

Increasing the Utility of Quantitative Empirical Studies for Meta-analysis

Heidi Lam & Tamara Munzner

University of British Columbia

Four Recommendations to Increase the Utility of Quantitative Empirical Studies

- Quantitative empirical studies are valuable tools in visualization evaluation, but may not be usable in meta-analyses
 - Meta-analyses may focus on different aspects of individual studies and have very different goals
 - Comparison between studies require common ground
- Derived **4 recommendations** based on our experience in a previous systematic review to make empirical studies more amenable to meta-analyses:
 1. Use comparable interfaces
 2. Capture usage patterns
 3. Isolate interface factors
 4. Report study details

Recommendation 1: Use Comparable Interfaces

- Identify and match key components in the interface to enable comparison between interface **factors** for single-factor studies
- Paper proposed five key components, focus on one in this talk:
 1. Basic Visual Element
 2. **Information content**
 3. Levels of display
 4. Levels in data
 5. Interaction complexity

Recommendation 1: Use Comparable Interfaces

Information content: what is displayed on the interface

- In our review, we wanted to know how different **spatial data-level arrangements** affect interface use:

- F+C (embedded)
- O+D (separate)

- The two interfaces also differed in content:

- F+C uses a **dynamic** algorithm to determine content readability
- O+D has **static** headers/subheaders

- Unable to tease out effects of data-level arrangements in time & accuracy results

Focus+Context (F+C)

Executable Object Modeling with Statecharts
David Harel & Eran Gery, Computer, JULY 1997, 30 no. 7, 31-42

Statecharts, popular for modeling system behavior in the structural analysis paradigm, are part of a fully executable language set for modeling object-oriented systems. The languages form the core of the emerging Unified Modeling Language.

Models for the development of object-oriented systems should be behaviorally expressive and rigorous as well as intuitive and well structured. Thus, any modeling approach must be detailed and precise enough to produce fully executable models and permit the automatic synthesis of efficient code in languages such as C++.

Our current implementation framework is based on C++, which is natural given its status in the OO language community. However, this is more a matter of convenience, so that models contain actions and operations written directly in the implementation language. This, in turn, makes it relatively easy to plug in a framework based on another language, such as Ada, Smalltalk, Java, or even on a set-based language. [5] However, what programming language is chosen as the implementation framework has little bearing on our modeling and analysis approach. Rhapsody supports the modeling process in its entirety, so once we

Low-level region based on a degree-of-interest algorithm

Overview+Detail (O+D)

Executable Object Modeling with Statecharts
David Harel & Eran Gery, Computer, JULY 1997, 30 no. 7, 31-42

Statecharts, popular for modeling system behavior in the structural analysis paradigm, are part of a fully executable language set for modeling object-oriented systems. The languages form the core of the emerging Unified Modeling Language.

Models for the development of object-oriented systems should be behaviorally expressive and rigorous as well as intuitive and well structured. Thus, any modeling approach must be detailed and precise enough to produce fully executable models and permit the automatic synthesis of efficient code in languages such as C++.

Most OO modeling notations. Entirely different notations, how they interrelate, specify behavior.

However, modeling languages. With time is not well defined and full executability and automatic code synthesis is impossible. Adopting a richly expressive behavioral language like statecharts makes modeling easier, but requires great care in defining the way it integrates with the other parts of the model. Statecharts must capture not only the state of the object as a precondition to service requests, but also the dynamics of the object's internal behavior in responding to those requests and in maintaining relationships with other objects.

These issues are complicated and go beyond recommending a modeling approach or methodology—they are language design concerns, requiring rigorous mathematical underpinnings. Both syntax and semantics must be fully worked out: any possible combination of constructs must be clearly characterized as syntactically legal or illegal, and each legal combination must be given a unique and formal meaning.

To address these needs, we embarked on an effort to develop an integrated set of diagrammatic languages for object modeling, built around statecharts, and to construct a supporting tool that produces a fully executable model and allows automatic code synthesis. The language set includes two *constructive* modeling languages (languages containing the information needed to execute the model or translate it into executable code).

- *Object-model diagrams* specify system structure by identifying object classes and their multiplicities, object relationships and roles, and subclassing relationships.
- *Statecharts* describe system behavior. A statechart attached to a class specifies all behavioral aspects of the objects in that class.

Low-level view shows headers and subheaders

Proposed Approach to Recommendation 1: Follow-up Studies

- It may be difficult to study the entire system **and** follow our recommendation to ensure comparable interfaces
 - E.g., The degree-of-interest algorithm in Hornbæk & Frøkjær's 2001/3 document-reading study is part of the F+C interface
- It is also difficult to tease out factors in advance
 - E.g., Differing levels of data between the Montana and Washington maps in Hornbæk et al.'s 2002 Map-navigation study



Washington map



Montana map

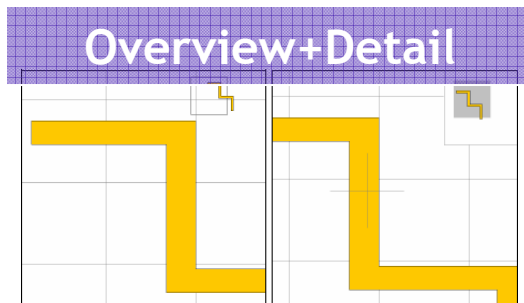
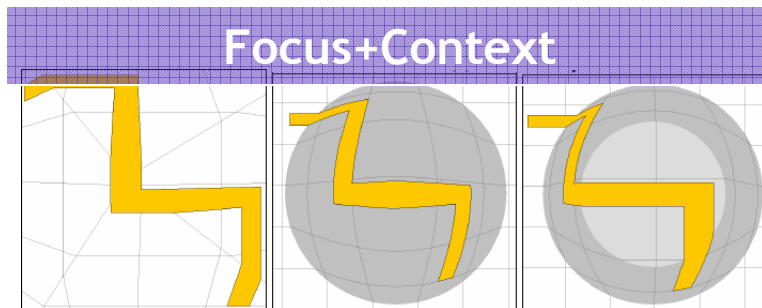
➤ Do **follow-up studies** to investigate identified factors

Recommendation 2: Capture Usage Patterns

- Provide usage patterns as insight into **how** an interface is used
 - separate from statistical results
- Despite being unable to use the time & accuracy results to tease out effects of data-level arrangement in Hornbæk and Frøkjær document-reading study, we **could** infer reading patterns from study paper
 - O+D better supports exploration since the overview offers navigation possibilities in a stable layout
 - F+C layout changes with user clicks
 - Insight into how spatial data-level arrangement affect use!
- Provide **observations** of interface use:
 - Participant strategies
 - Interface choice
 - Interactivity (e.g., with eye-gaze recordings, usability logs)

Recommendation 3: Isolate Interface Factors

- Completely cross all factors to allow isolating effects of a single factor in a multi-factor study



- Gutwin & Skopik's 2003 2D-steering study has at least 3 factors
- We were only interested in the effects of spatial data-level arrangement, but not the other two (effective steering path, interaction complexity)
- We were unable to isolate the effect of spatial data-level arrangement for our purpose of review, as the factors were not fully-crossed

- Do **follow-up studies** to look at a subset of the factors to limit the number of conditions

Recommendation 4: Report Study Details

- Chen and Yu (2000) recommended standardizations in reporting
 - Testing information
 - Statistical results
 - Descriptions of visual-spatial properties of information visualization systems
- We added a few based on our systematic review
 - Task instructions: to ascertain levels of clues provided
 - Interaction video/demo: to understand interaction complexity
- Use **online materials** as workaround for publication page limits
 - Online supplementary materials
 - Online project websites

Possible Outcomes:

Standardize Studies at a Re-usable level

- We proposed 4 recommendations to better empirical studies:
 1. Use comparable interfaces → Do follow-up studies
 2. Capture usage patterns → Provide observations
 3. Isolate interface factors → Do follow-up studies
 4. Report study details → Post online materials
- Doing so will hopefully:
 - Enable and encourage reviews and meta-analyses to capture existing knowledge
 - Enable building of interface-factor repositories at a re-usable level