- Is navigation efficient and straightforward?
- Has the WebApp architecture been designed to accommodate the special goals and objectives of WebApp users, the structure of content and functionality, and the flow of navigation required to use the system effectively?
- Are components designed in a manner that reduces procedural complexity and enhances correctness, reliability, and performance?

Today, each of these questions can be addressed qualitatively,[16] but a validated suite of metrics that would provide quantitative answers does not yet exist.

Metrics for WebApp design are in their infancy, and few have been validated widely. The interested reader should see [IVO01] and [MEN01] for a sampling of proposed WebApp design metrics.

SOFTWARE TOOLS

Technical Metrics for WebApps

Objective: To assist Web engineers in developing meaningful WebApp metrics that provide insight into the overall quality of an application.

Mechanics: Tool mechanics vary.

Representative Tools[17]

Netmechanic Tools, developed by Netmechanic (www.netmechanic.com), is a collection of tools that help to improve Web-site performance, focusing on implementation-specific issues.

NIST Web Metrics Testbed, developed by The National Institute of Standards and Technology (zing.ncsl.nist.gov/WebTools/), encompasses the following collection of useful tools that are available for download:

Web Static Analyzer Tool (WebSAT)—checks web page HTML against typical usability guidelines.

Web Category Analysis Tool (WebCAT)—lets the usability engineer construct and conduct a Web category analysis.

Web Variable Instrumenter Program (WebVIP)—instruments a Web site to capture a log of user interaction.

Framework for Logging Usability Data (FLUD)—implements a file formatter and parser for representation of user interaction logs.

VisVIP Tool—produces a 3D visualization of user navigation paths through a Web site.

TreeDec—adds navigation aids to the pages of a Web site.

19.12 SUMMARY

The quality of a WebApp—defined in terms of usability, functionality, reliability, efficiency, maintainability, security, scalability, and time-to-market—is introduced during design. To achieve these quality attributes, a good WebApp design should exhibit simplicity, consistency, identity, robustness, navigability, and visual appeal.

Interface design describes the structure and organization of the user interface. It includes a representation of screen layout, a definition of the modes of interaction, and a description of navigation mechanisms.

---

16 See Chapter 16 (Section 16.4) and Section 19.1.1 for a qualitative discussion of WebApp quality.

17 Tools noted here do not represent an endorsement, but rather a sampling of tools in this category.

Aesthetic design, also called graphic design, describes the "look and feel" of the WebApp and includes color schemes, geometric layout, text size, font and placement, the use of graphics, and related aesthetic decisions. A set of graphic design guidelines provides the basis for a design approach.

Content design defines the layout, structure, and outline for all content that is presented as part of the WebApp and establishes the relationships between content objects. Content design begins with the representation of content objects, their associations, and relationships. A set of browsing primitives establishes the basis for navigation design.

Architecture design identifies the overall hypermedia structure for the WebApp and encompasses both content architecture and WebApp architecture. Architectural styles for content include linear, grid, hierarchical, and network structures. WebApp architecture describes an infrastructure that enables a Web-based system or application to achieve its business objectives.

Navigation design represents the navigational flow between content objects and for all WebApp functions. Navigation is defined by describing a set of navigation semantic units. Each unit is composed of ways of navigation and navigational links and nodes. Navigation syntax mechanisms are used for effecting the navigation described as part of the semantics.

Component design develops the detailed processing logic required to implement WebApp functional components. Design techniques described in Chapter 11 apply to the engineering of WebApp components.

Patterns for WebApp design encompass generic design patterns that apply to all types of software and hypermedia patterns that are especially relevant for WebApps. Architecture, navigation, component, presentation, and behavior/user design patterns have been proposed.

The Object-Oriented Hypermedia Design Method (OOHDM) is one of a number of methods proposed for WebApp design. OOHDM suggests a design process that includes conceptual design, navigational design, abstract interface design, and implementation.

Design metrics for Web engineering are in their infancy and have yet to be fully validated. However, a variety of measures and metrics have been proposed to address each of the design activities discussed within this chapter.

REFERENCES

[AME96] Amento, B., et al., "Fitt's Law," *CS 5724: Models and Theories of Human-Computer Interactions,* Virginia Tech, 1996, available at http://ei.cs.vt.edu/ ~cs5724/g1/.

[BAG01] Baggerman, L., and S. Bowman, *Web Design That Works,* Rockport Publishers, 2001.

[BUS96] Buschmann, F., et al., *Pattern-Oriented Software Architecture,* Wiley, 1996.

[CAC02] Cachero, C., et al., "Conceptual Navigation Analysis: a Device and Platform Independent Navigation Specification," *Proc. 2nd Intl. Workshop on Web-Oriented Technology,* June 2002, download from www.dsic.upv.es/~west/iwwost02/papers/cachero.pdf.