Context-Dependent Behavior is Becoming Increasingly Important for a Wide Range of Application Domains, from Pervasive Computing to Common Business Applications. Unfortunately, Mainstream Programming Languages Do Not Provide Mechanisms That Enable Software Entities to Adapt Their Behavior Dynamically to the Current Execution Context. This Leads Developers to Adopt Convoluted Designs to Achieve the Necessary Runtime Flexibility. We Propose a New Programming Technique Called Context-Oriented Programming (COP) Which Addresses This Problem. COP Treats Context Explicitly, and Provides Mechanisms to Dynamically Adapt Behavior in Reaction to Changes in Context, Even After System Deployment at Runtime. In This Paper, We Lay the Foundations of COP, Show How Dynamic Layer Activation Enables Multi-Dimensional Dispatch, Illustrate the Application of COP by Examples in Several Language Extensions, and Demonstrate That COP is Likely Independent of Other Commitments to Programming Style.

1. Introduction

Contextual Information Is Playing an Increasingly Important Role for Applications and Services Ranging from Those That Are Location-Based to Those That Are Situation-Dependent or Even Deeply Personalized. While Context-Awareness Is Already an Integral Part of Regular Business Applications, It Is Becoming Even More Critical for Mobile and Ubiquitous Computing, Where Devices Must Adapt Their Behavior to the Services Available in Their Current Environment.

Despite the Fact That Context Is Clearly a Central Notion to an Emerging Class of Applications, There Is Little Explicit Support for Context Awareness in Mainstream Programming Languages and Runtime Environments. This Makes the Development of These Applications More Complex Than Necessary. In This Paper We Argue the Need for a New Programming Approach, Called Context-Oriented Programming (COP), Which Treats Context Explicitly and Makes It Accessible and Manipulable by Software.

One Difficulty in Proposing a Concrete COP Language Is That Context Covers a Wide Range of Concepts Ranging From Domain-Specific to Technology-Dependent Attributes, and Including Properties That May Be Spatial or Temporal, or Even Based in Hardware or Software. We See Personalization, Sharing, Location-Awareness, Ubiquity, and Context-Awareness as Examples of Such Attributes.


Context-Oriented Programming for Wireless Sensor Networks

Wireless Sensor Networks Are Inherently Environment Dependent

Context Is a Representation of the Environment

Context-Oriented Programming Is Using Context as a Building Block

Example: UAV Controller

- Should save energy but provide a high quality of service
- Should transmit data if the base station is in the reach
- Should locally log data in infrastructure-less situations

Context-Oriented nesC (ConesC)

Example

Mapping

Evaluation:

- Two applications
- The same functionality
- TelosB platform

Ongoing Work

- Implementation of the translator
- Evaluation on real applications

UBICOMP13 - Zurich (Switzerland), September 8th-12th