Design Approach Review

GUARDIAN

(General User AleRt Display pANel)

AWIPS Build OB4

MDL

Tom Filiaggi
July, 2003

GUARDIAN

AGENDA

- Overview
- Data Flow
- Software Design
- (Data Handling)
- Graphical Interface
- Installation
- Performance
- Testing
- Hardware Resources
- Responsible
- Documentation
- Schedule

GUARDIAN

Overview

- Background
  - Currently, the AWIPS users can receive messages from various software in various forms with little control.
  - Anecdotal evidence suggests that users are rather unhappy about how AWIPS software developers have chosen to notify them of various real-time issues, including:
    - Overwhelming audio
    - Unnecessarypop-ups
    - Lack of “pertinent” information (which is, of course, subjective)
  - The growing number of meteorological monitors is quickly running out of space on the D2D tool bar.

GUARDIAN

Overview

- Implementation Strategy
  - Develop a new communicator to allow the user to filter what software messages they receive.
  - Develop a new communicator to allow the user to define how they wish to be notified of messages with varying sources and priorities.
  - Have a persistent process running on each workstation that utilizes general AWIPS IPC methods.

GUARDIAN

Overview

- The General User Alert Display Panel will:
  - Provide a vehicle to communicate messages from software to the user. These messages can be:
    - From meteorological monitoring software
    - System health alerts
    - Radar ingest messages
    - Based on Informix Triggers
    - Messages from **any** other client application . . .
  - Provide methods to allow users to configure how these messages get communicated: blink, beep, pop-up, or even run an action script or play a sound file.

GUARDIAN

Customization

Set Up

Customization

GUARDIAN GUI

Default Customization

Furnished Customization #1

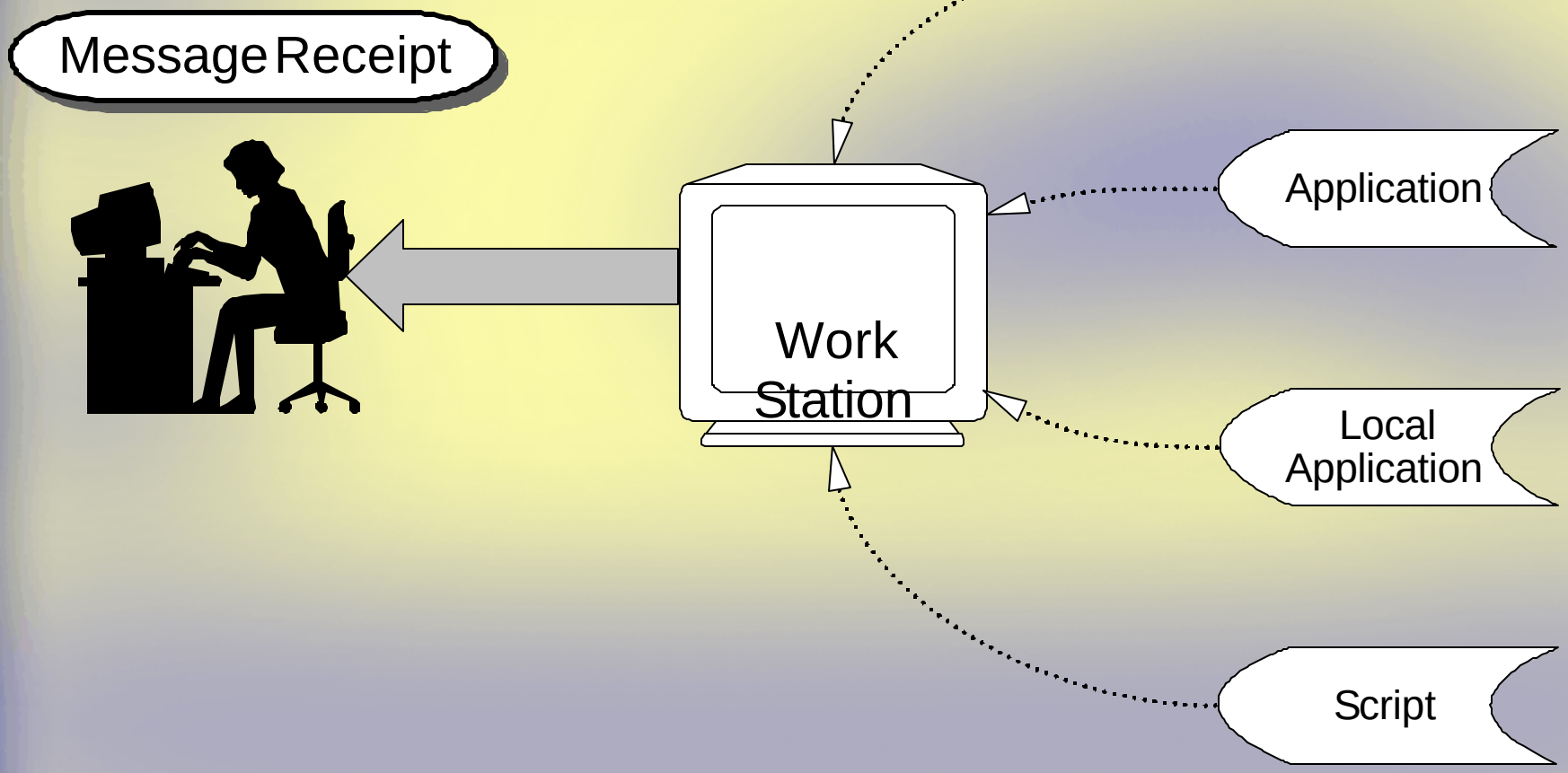
Furnished Customization #2

Furnished Customization #3

Used Customization

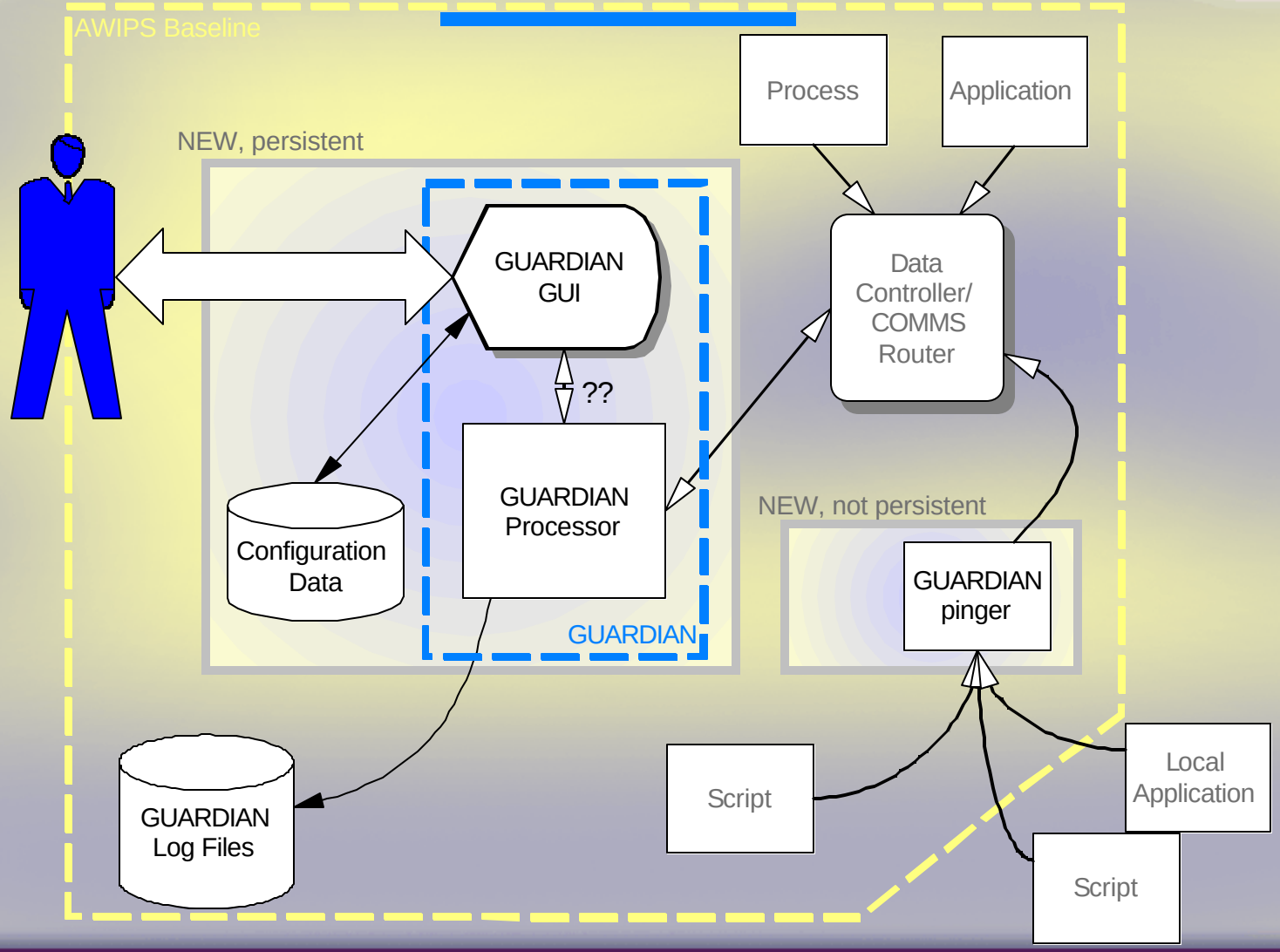
GUARDIAN

Message Receipt



GUARDIAN

Data Flow



GUARDIAN

Software Design

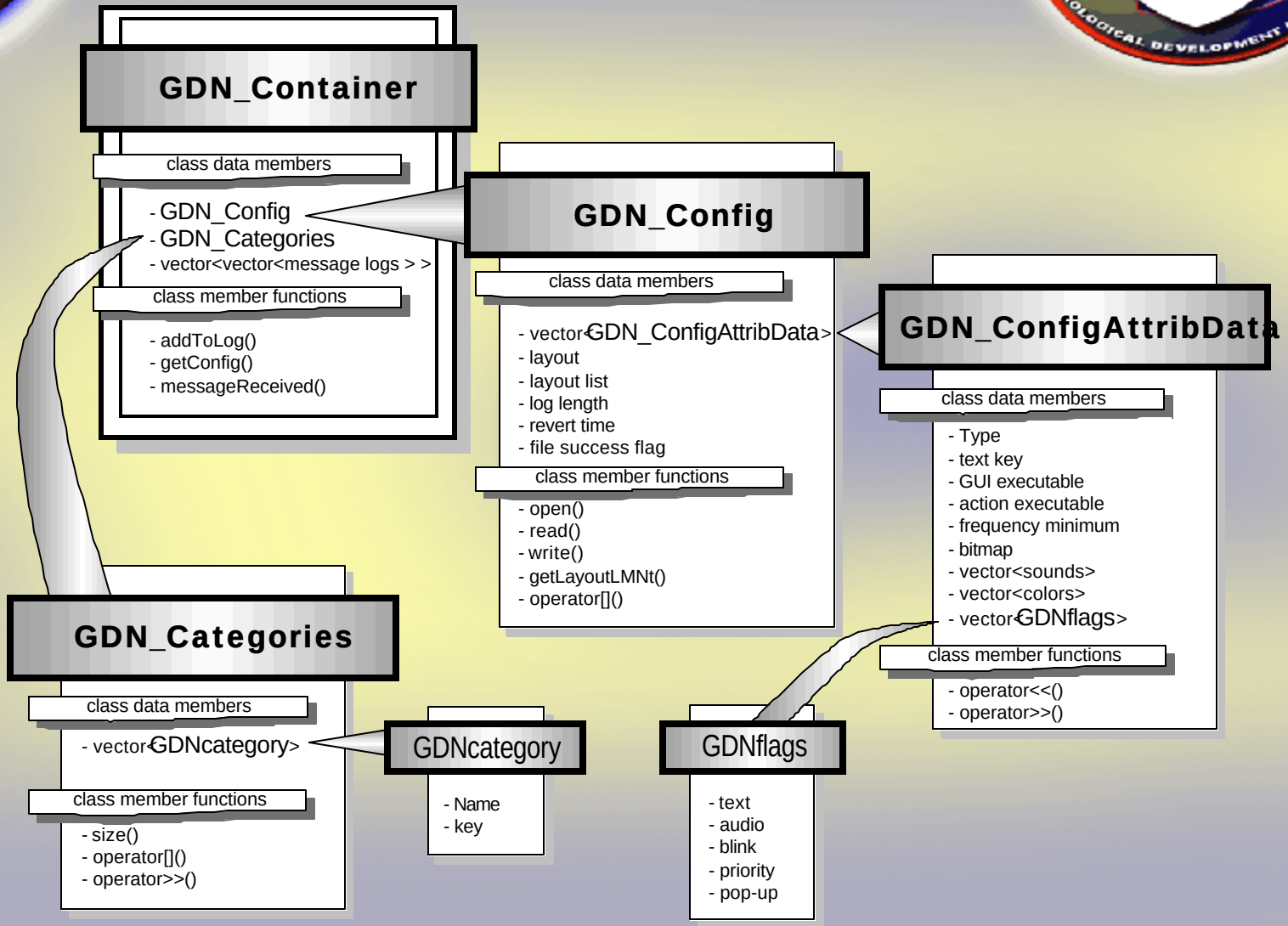
- - Programming Languages used
  - C++
  - TCL/TK
  - Perhaps shell script
- COTS usage
  - TCL
- Machine Specific Dependencies
  - LINUX (no plans for HP compliance)
- Service APIs Required
  - none

GUARDIAN

Software Design

- Use C++ foundation, with Tcl Interpretor.
- Use AWIPS IPC: registration and message handling.
- Use comprehensive configuration data, to provide user-friendly flexibility.
- Attempt to use one executable which will handle all of the above items.
- Run on each workstation to allow workstation-specific configuration.

GUARDIAN


GDN_Container
class data members
- GDN_Config
- GDN_Categories
- vector<vector<message logs > >
class member functions
- addToLog()
- getConfig()
- messageReceived()
GDN_Config
class data members
- vector<GDN_ConfigAttribData>
- layout
- layout list
- log length
- revert time
- file success flag
class member functions
- open()
- read()
- write()
- getLayoutLMNt()
- operator[]()
GDN_ConfigAttribData
class data members
- Type
- text key
- GUI executable
- action executable
- frequency minimum
- bitmap
- vector<sounds>
- vector<colors>
- vector<GDNflags>
class member functions
- operator<<()
- operator>>()
GDN_Categories
class data members
- vector<GDNcategory>
class member functions
- size()
- operator[]()
- operator>>()
GDNcategory
- Name
- key
GDNflags
- text
- audio
- blink
- priority
- pop-up

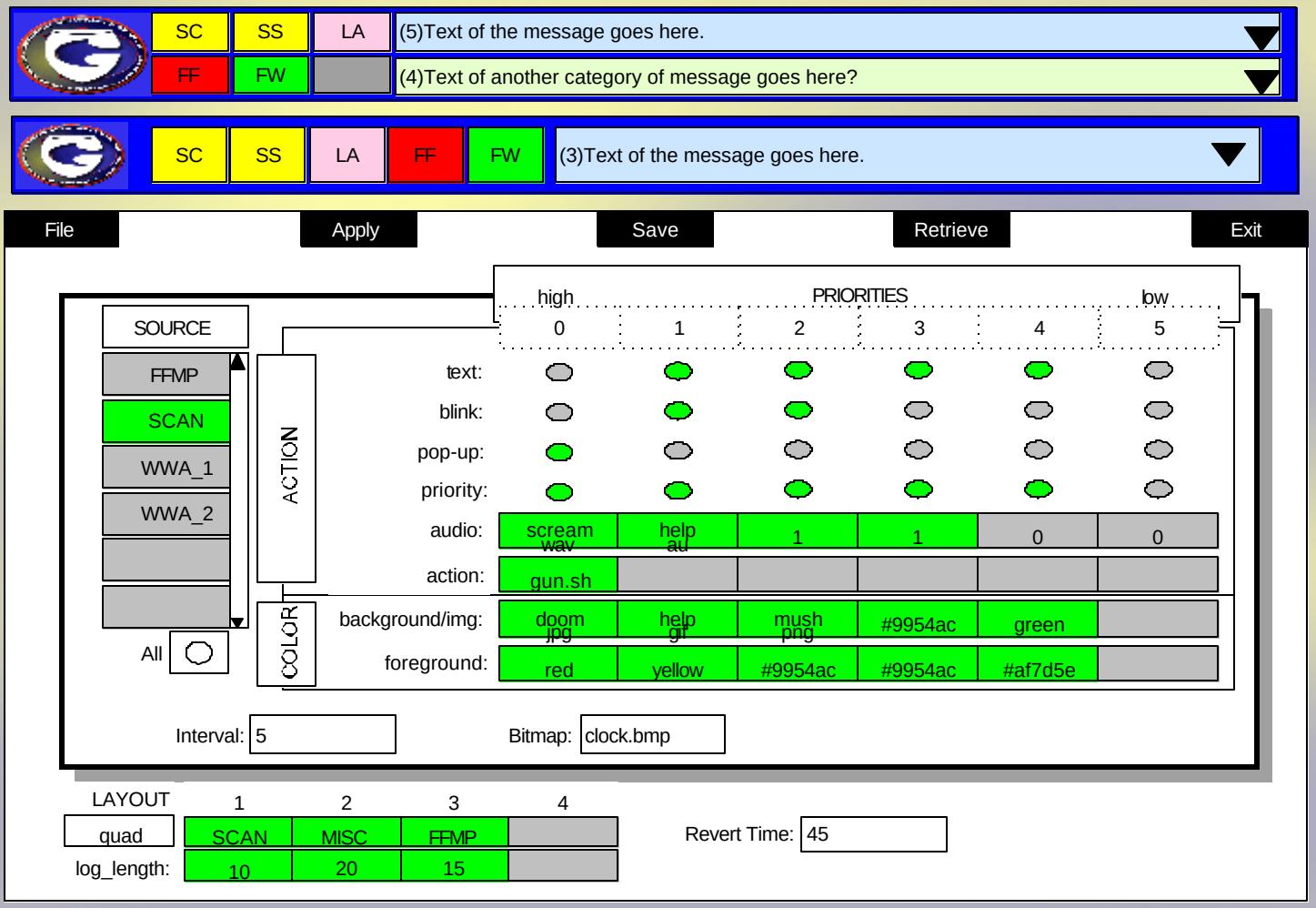
GUARDIAN

Graphical User Interface

- **Must** remain “on top”, above all other windows!
- **Must** occupy minimal screen-estate, due to ‘on top’ status!
- Provide flexible methods to configure how messages get conveyed to the user.
- Provide customizable GUI appearance.
- Provide methods to store and retrieve configurations.
- Provide a small number of default settings, to ensure ‘out of the box’ use. Examples of such modes could be:
  - “Severe Weather”
  - “Forecast”
  - “System Monitor”
  - “Text WorkStation”

GUARDIAN

Graphical User Interface



GUARDIAN

Installation

- **Need to start GUARDIAN upon log-in!** Log-in files will need to be edited.
- **No** changes to National Metadata files
- **No** changes to site-modified metadata files
- **No** changes to runtime setup files
- **No** expected cron usage
- **No** new runtime disk partitioning / directories anticipated being created
- New runtime metadata files created: Configuration files
- **No** core runtime system services changes anticipated
- **No** new COTS / freeware runtime packages anticipated

GUARDIAN

Performance

- Assessment of performance
  - Minimal new CPU load
- Assessment of shared services with new design
  - Expected increase in IPC Comms traffic.
- **No** anticipated issues with algorithmic performance
- Disk I/O usage to be determined
- **No** anticipated use of remote shell, rcp, or other such system calls

GUARDIAN

Testing

- Internal Testing: NHDW, NHDA
- Alpha test: Yet to be determined, but have had several volunteer WFOs.

GUARDIAN

Hardware/Resource Usage Design

- **No** new Hardware or mods to existing hardware required by this item
- Additional Disk Space
  - Log space will depend on degree of use. May be significant.
- **No** anticipated use of Omniback/ tapedrive
- **No** anticipated use of the WAN
- **No** anticipated use of the SBN
- **No** potential problematic use of special hardware resources

GUARDIAN

Assignment of Responsible Individuals

- Main Developers
  - Tom Filiaggi - MDL: Lead

GUARDIAN

Schedule

- Prototype Preparation
  - Fall, 2003
- User Interface Review & Alpha Testing
  - November, 2003
- End of Development
  - January, 2004