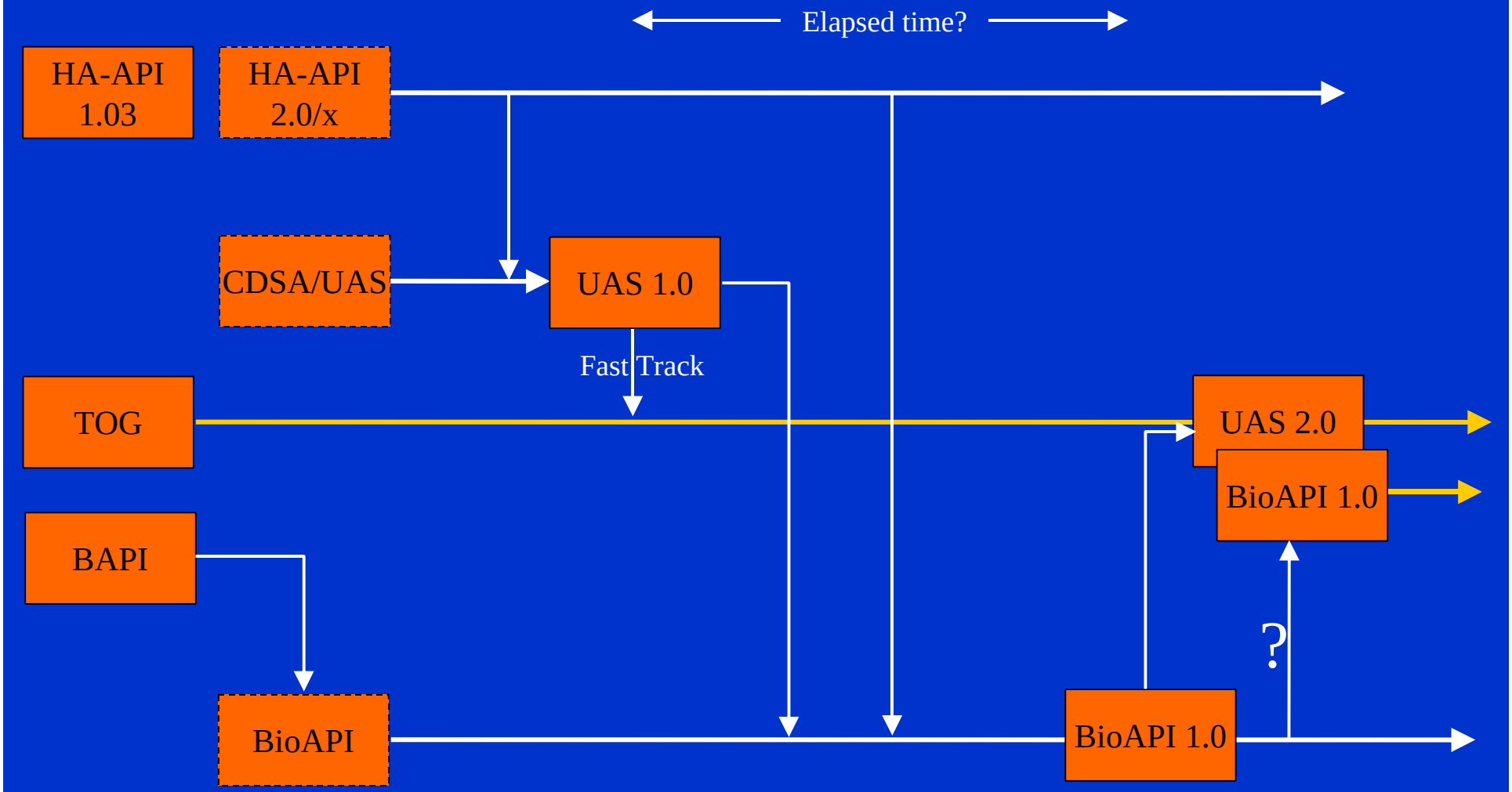
CDSA/UAS
HA-API and BioAPI

John.H.Wilson@Intel.com
Security Technology Lab
Platform Security Division
April 28th, 1999

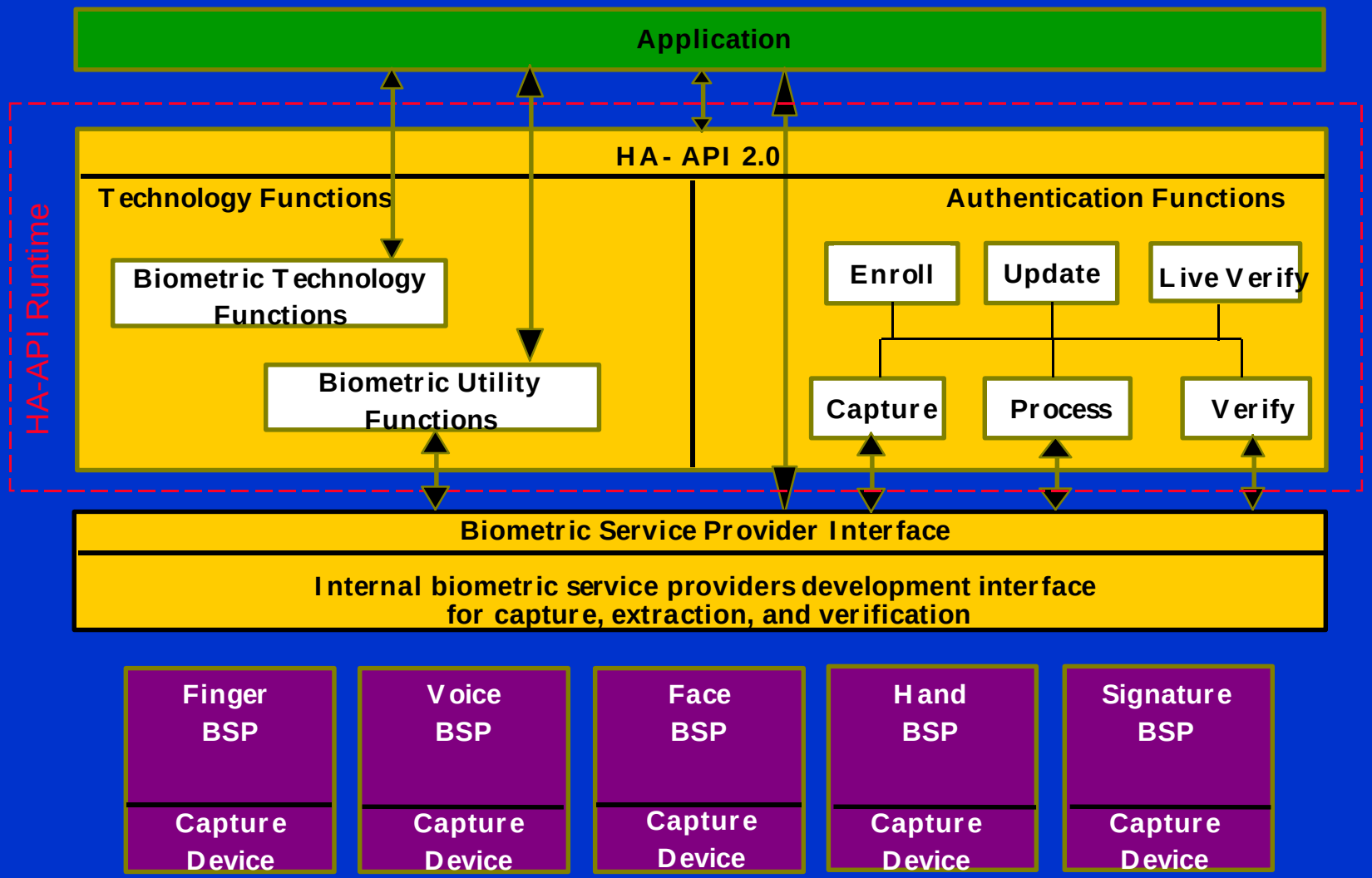
Agenda

- Merging Standards
- HA-API 2.0
- BioAPI
- CDSA + UAS
- UAS
- Protection with PINs and Pass-phrases
- Protection with Biometrics

Standards Roadmap



HA-API Architecture



HA-API Functions (2.0)

Biometric Technology Functions

- ***EnumBioTechnology*** - Enumerates the biometric technologies installed on a system and available for use by the application.
- ***GetBioTechnology*** - Initializes a biometric technology.
- ***ReleaseBioTechnology*** - Frees the handle created by GetBioTechnology.

Biometric Authentication Functions

- ***HAAPICapture*** - Captures raw biometric data.
- ***HAAPIProcess*** - Processes the raw biometric data and extracts a unique Biometric Identifier Record (BIR).
- ***HAAPIVerify*** - Performs a 1:1 verification match against two BIRs.
- ***HAAPILiveVerify*** - Intended for use by technologies that perform continuous capture, process, and verifyication until a match is found or a time-out is reached.
- ***HAAPIEnroll*** - Captures and processes raw biometric data, encapsulating the entire process of enrollment.
- ***HAAPIUpdate*** - performs adaptation, reenrollment, or updates to previously enrolled/stored biometric data.
- ***HAAPIIdentify*** - Placeholder for future 1:N search/match function.

Biometric Utiltity Functions

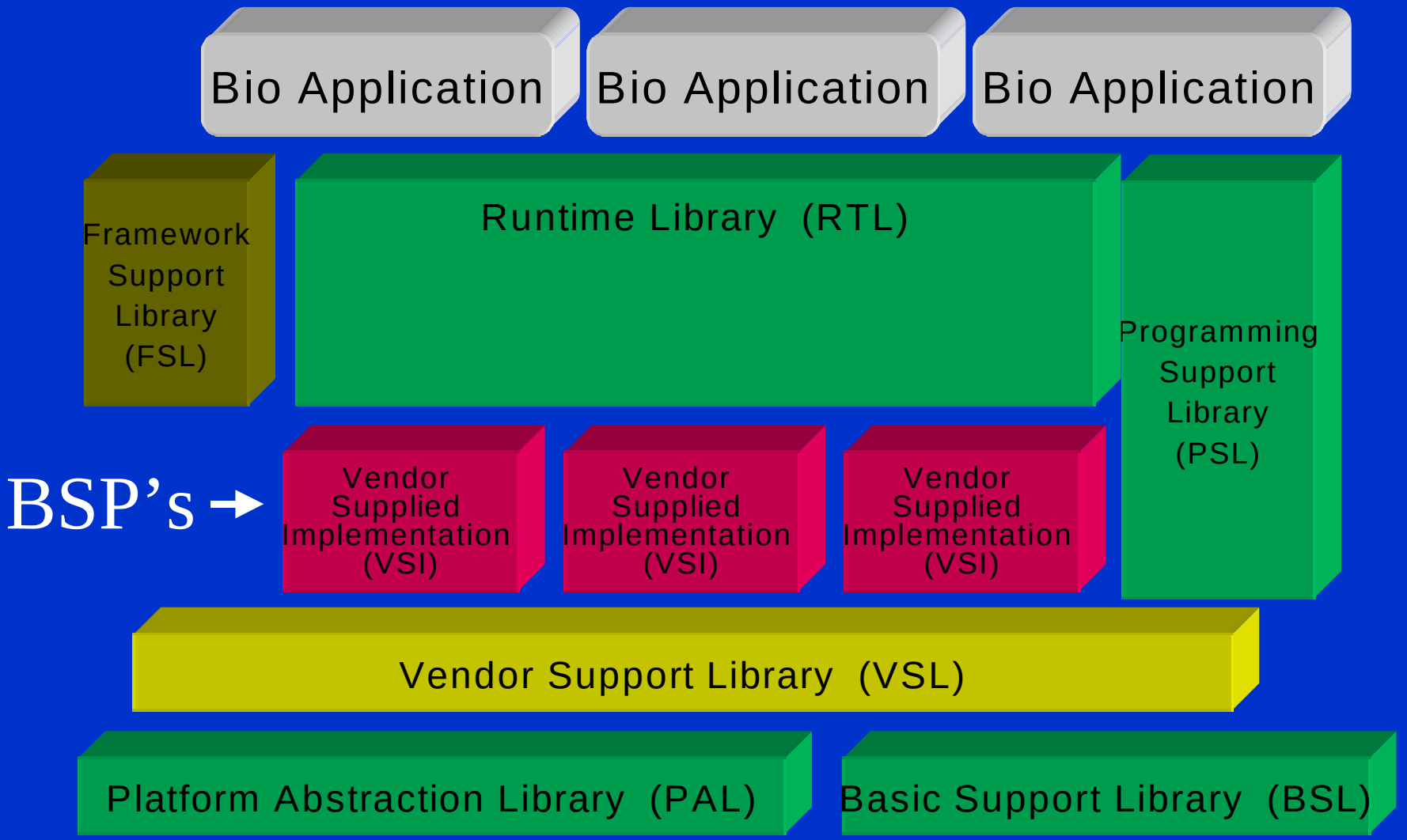
- ***HAAPIInformation*** - Provides an interface for the application developer to get and set information specific to a biometric technology.
- ***HAAPIFree*** - Deallocates memory used by a previous biometric function.
- ***HAAPIBioProperties*** - Used to read and set parameters specific to a biometric and its input devices.

Biometric DB Functions *

- ***HAAPIDBAdd***
  - Add one or more users data to the BSP DB.
- ***HAAPIDBUpdate***
  - Update user data in the BSP DB.
- ***HAAPIDBDelete***
  - Delete one or more users from the BSP DB.
- ***HAAPIDBRetrieve***
  - Retrieve user data from the BSP DB.

* Proposed for Ver 2.1

BioAPI Framework



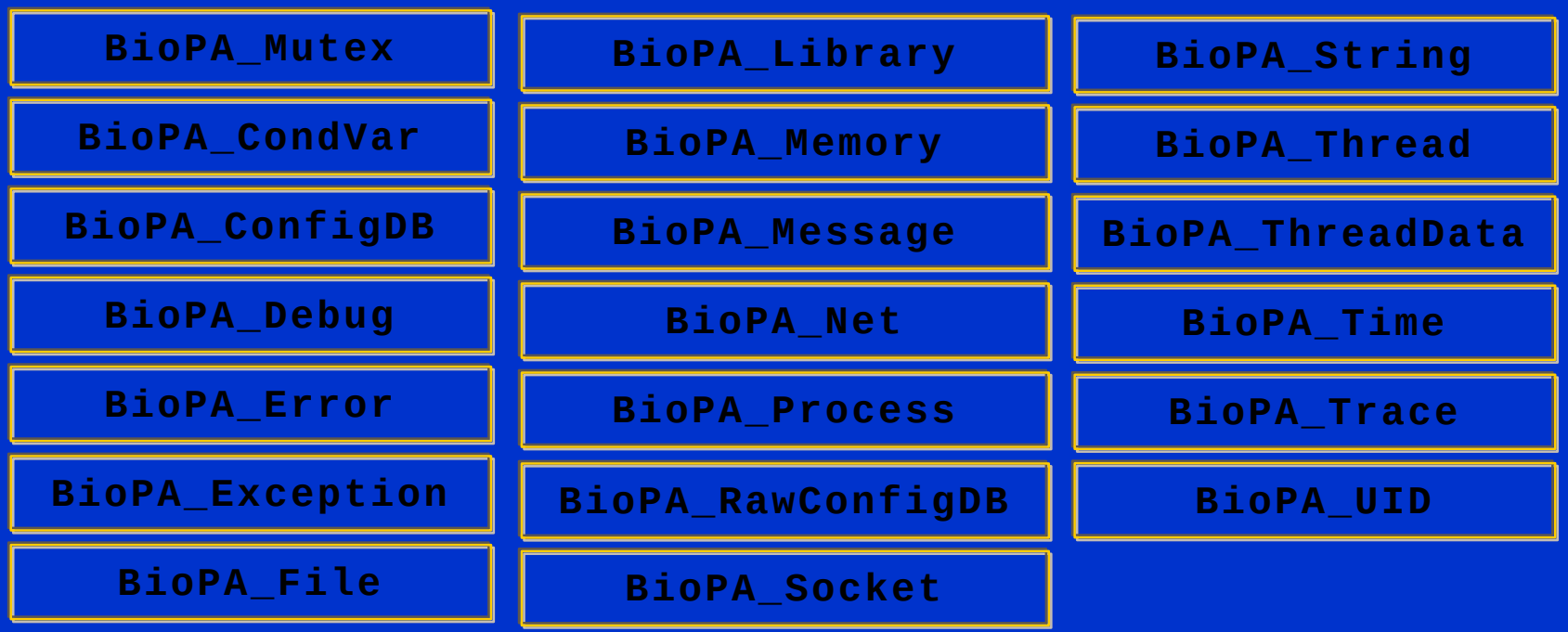
Programming Support Library (PSL)

BioByteArray	BioObjectInfo
BioCollection	BioObjectServices
BioDebug	BioSpace
BioError	BioString
BioHandle	BioTrace
BioInterface	BioUID
BioMemory	BioVendor

Vendor Support Library (VSL)

BioVS_Config	BioVS_Object	BioVS_Trace

Platform Abstraction Library (PAL)



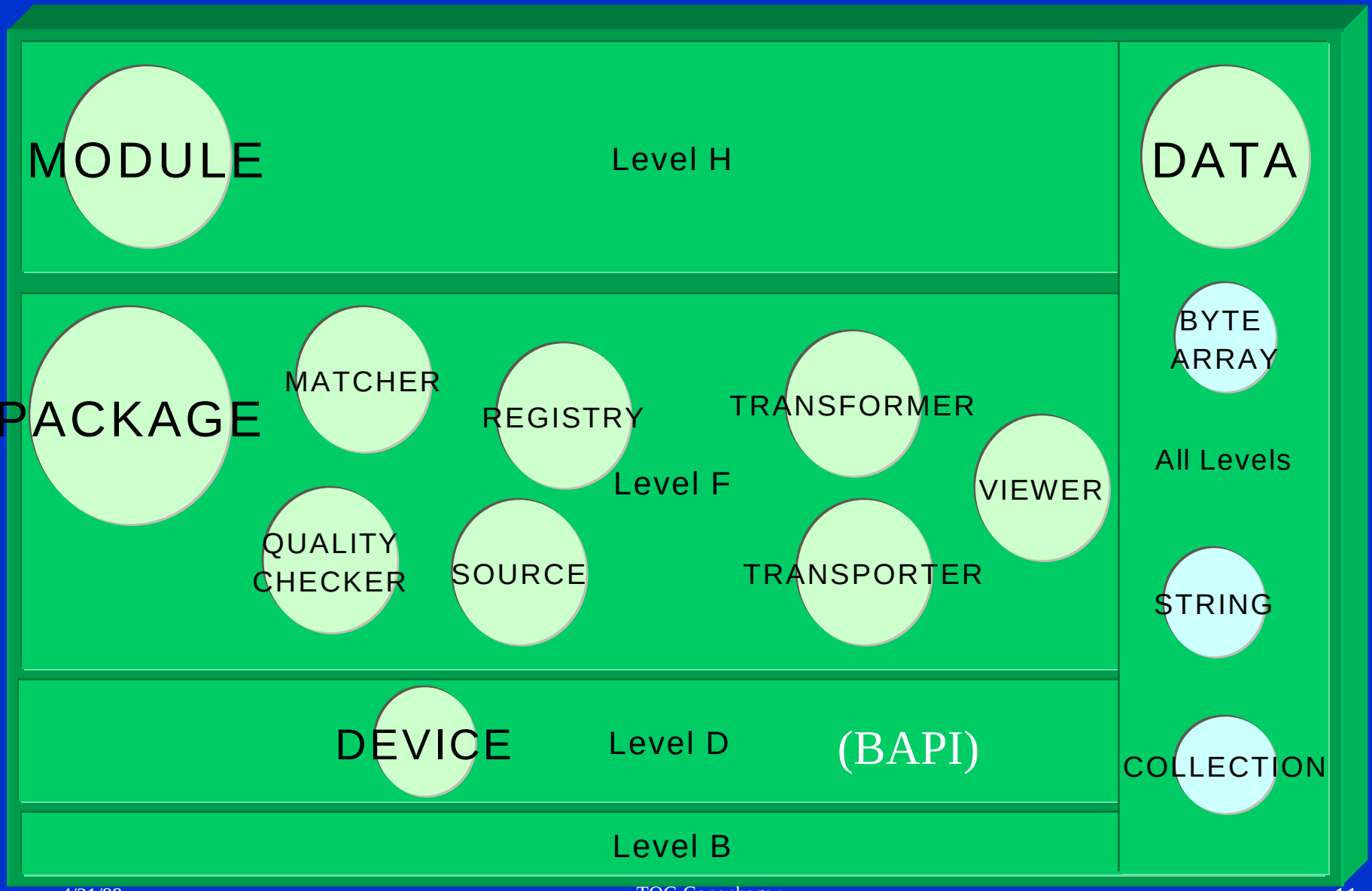
Basic Support Library (BSL)

- BioBS_ByteArray
- BioBS_Code
- BioBS_HandleTable
- BioBS_Interface
- BioBS_Level
- BioBS_Module
- BioBS_Object
- BioBS_String
- BioBS_UIDTable

Framework Support Library (FSL)

- BioFS_Data
- BioFS_Handle
- BioFS_Implementation
- BioFS_Object
- BioFS_Package
- BioFS_Space

The BioAPI Objects (implemented by BSP)



BioAPI

HA-API 2.0 functionality + Security + ….

HA-API AL

BioAPI Runtime

BioAPI BSP

(black box)

BioAPI BSP

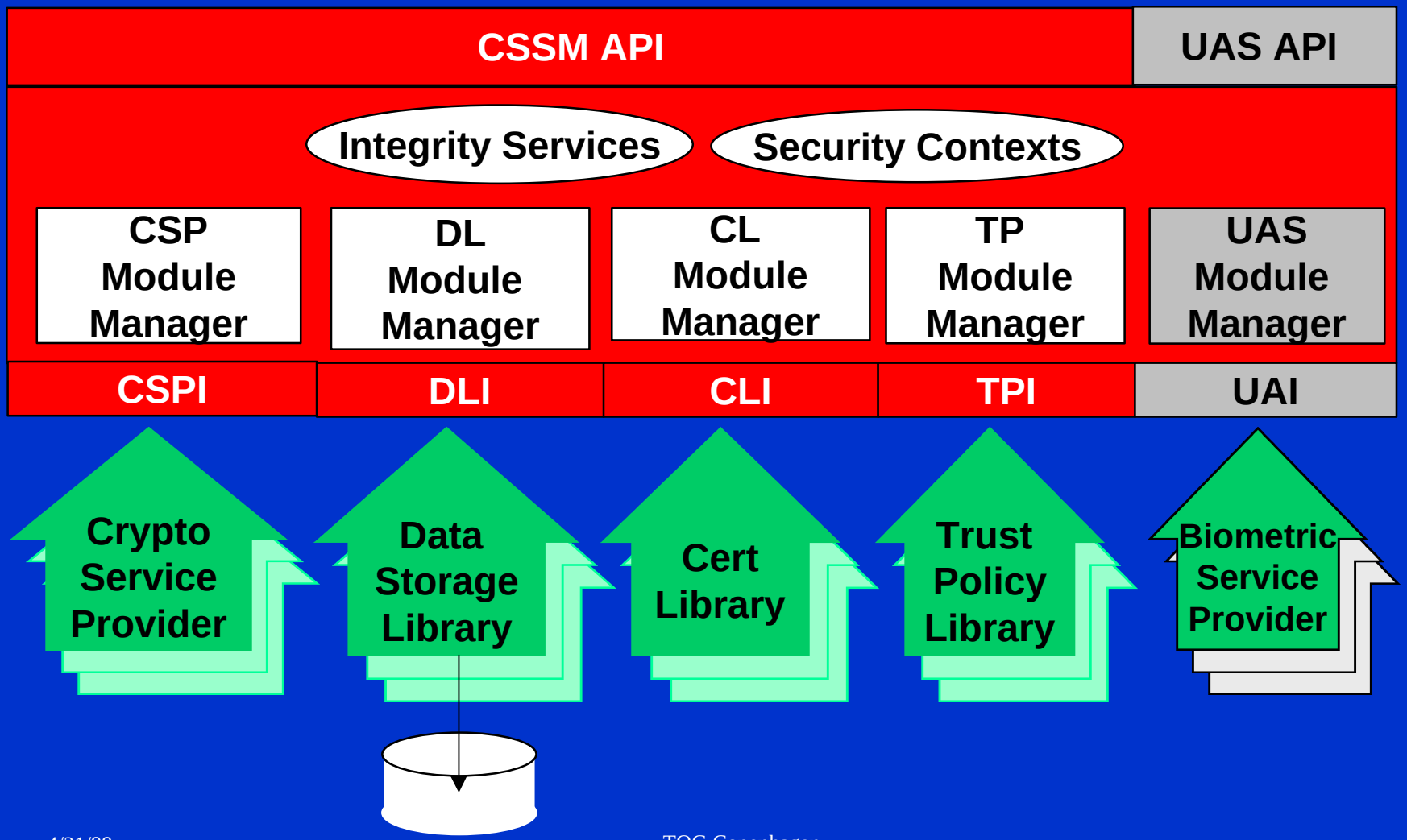
Using BioAPI layered architecture

HA-API AL

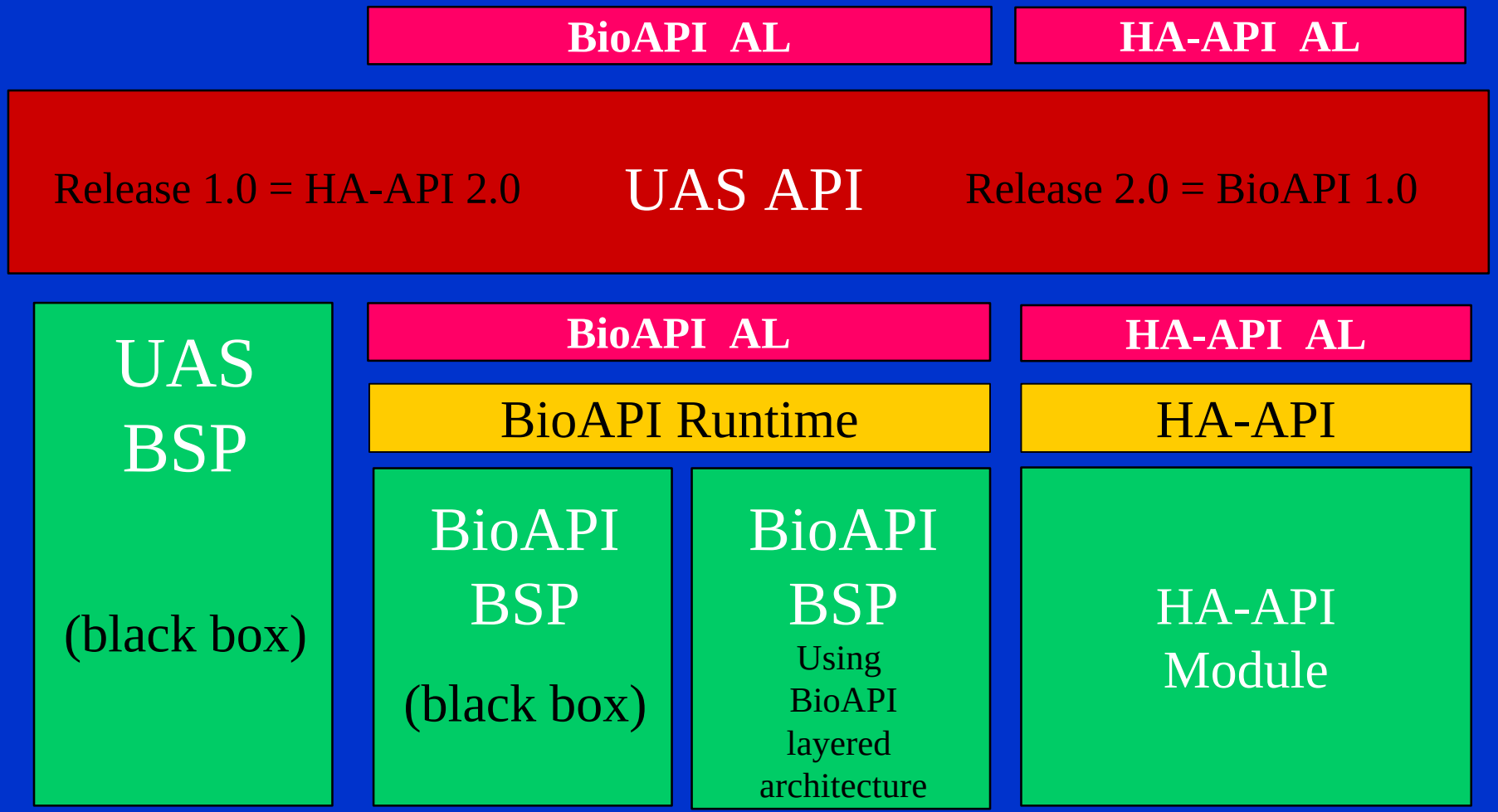
HA-API

HA-API Module

CSSM + User Authentication Service



UAS


BioAPI AL
HA-API AL
Release 1.0 = HA-API 2.0
UAS API
Release 2.0 = BioAPI 1.0
UAS BSP
(black box)
BioAPI AL
BioAPI Runtime
BioAPI BSP
(black box)
BioAPI BSP
Using BioAPI layered architecture
HA-API AL
HA-API
HA-API Module

Why UAS?..... Why TOG?

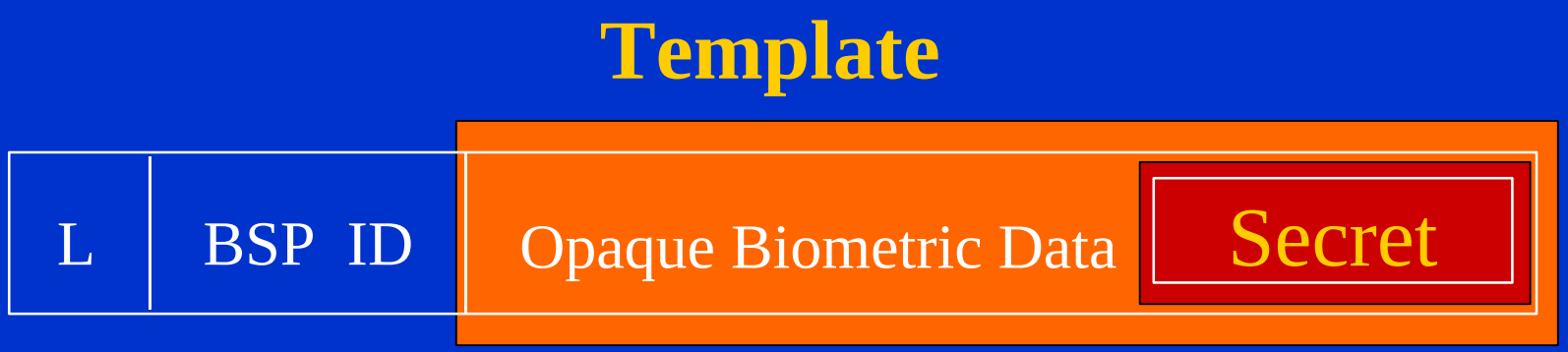
- Integrity of Implementation
  - protection of authentication data
  - allowing protection of cryptographic secrets
- Ubiquity
  - availability of framework to applications
- Control of Conformance
- Long-term maintenance of Standard
- TOG marketing

Use of Credentials in CSPI

- Login to Token
  - Login(h, **Credentials**,....)
- Key Protection
  - CreateKey (h, AlgorithmID, **Credentials**, key,....)
  - Sign (h, AlgorithmID, **Credentials**, key,....)
  - Encrypt (h, AlgorithmID, **Credentials**, key,....)
  - etc etc

Credentials can be "biometric"

Biometric-Wrapped Secrets



Secret — Wrapped Cryptographic Key
or PIN for CSP/SmartCard Login

Opaque Biometric Data plus Secret
Signed with BSP's Private Key
Encrypted with BSP's Public Key

BSP ID — Identifies Biometric Service Provider (BSP)

Examples

CreateKey (h, ... “biometric”, keyid,....)

Enroll (. . , Secret, Template)

Secret is wrapped key.

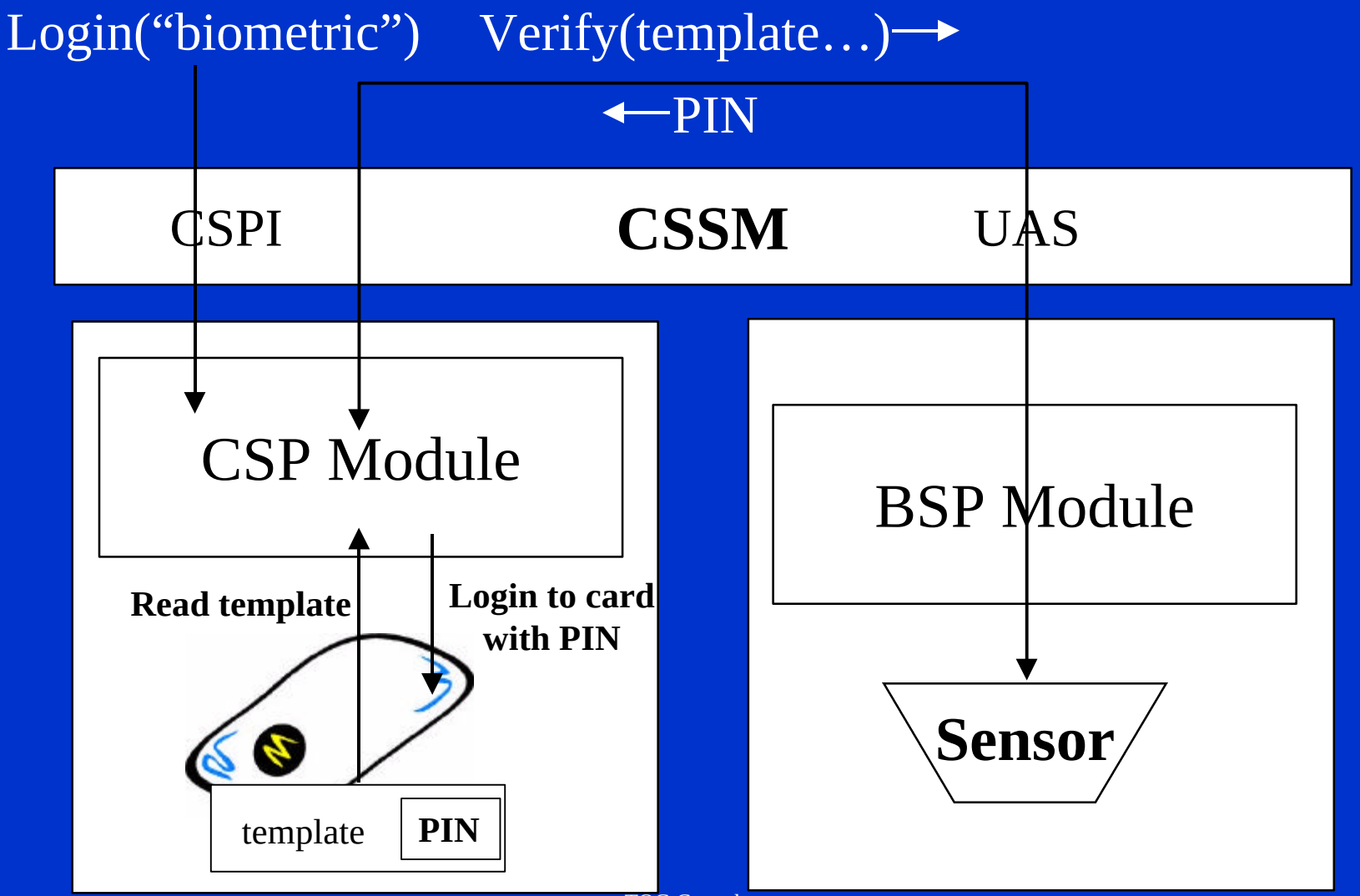
UseKey (h, ... “biometric”, keyid,....)

Releases key if User’s Biometric matches

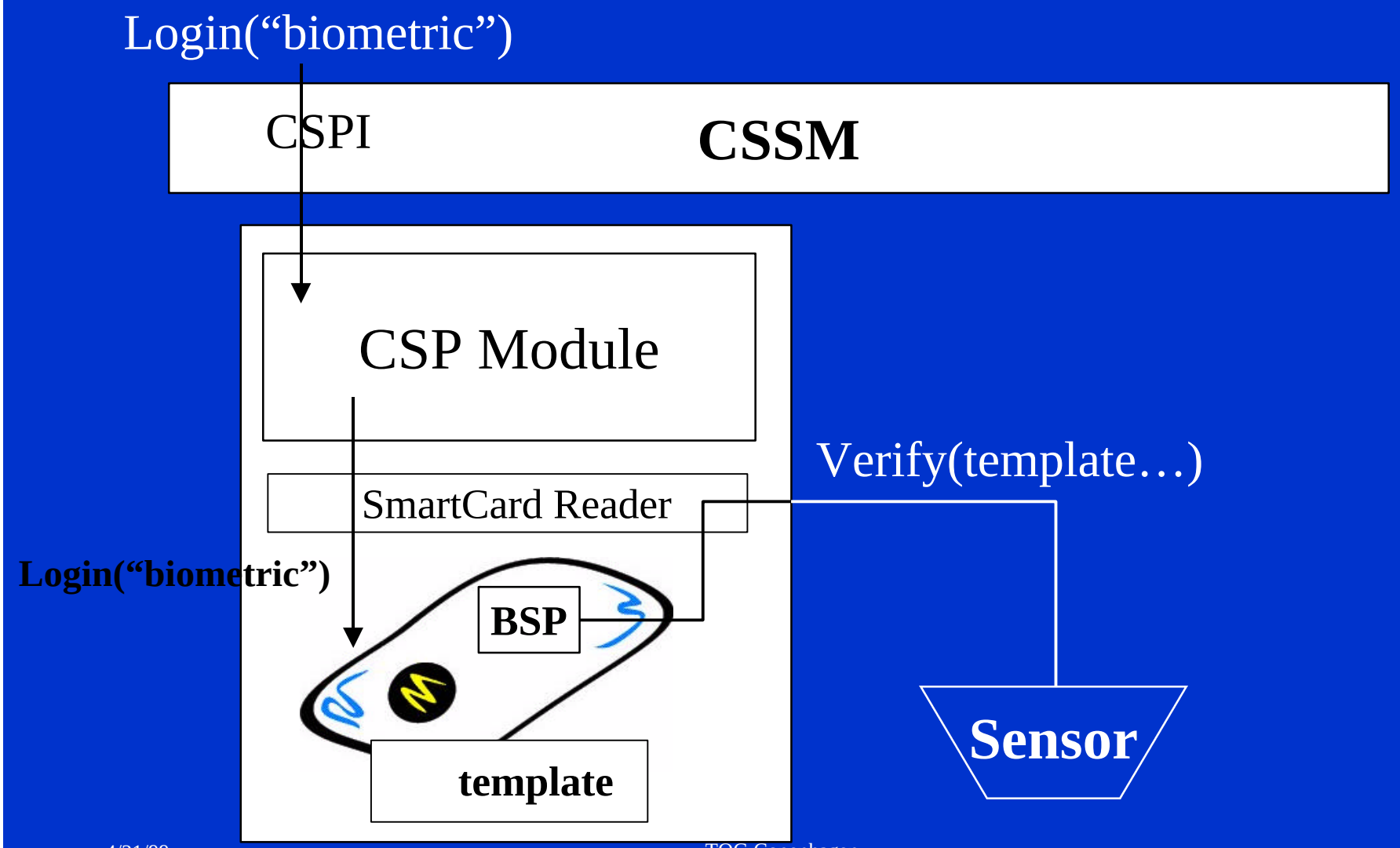
Verify (. . , Template, Secret, . . .)

Secret from template if Verify successful

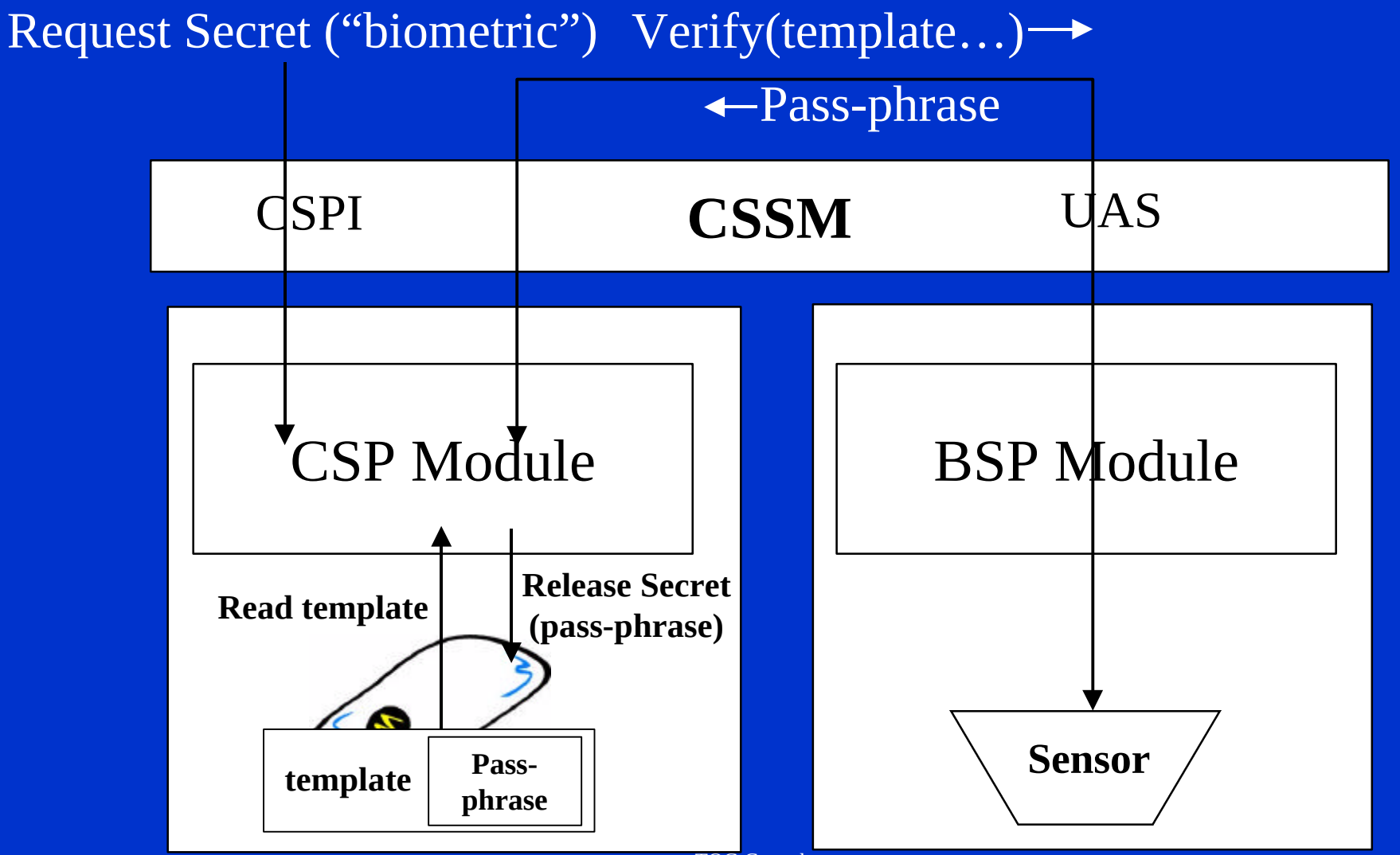
SmartCard Login (evolutionary)



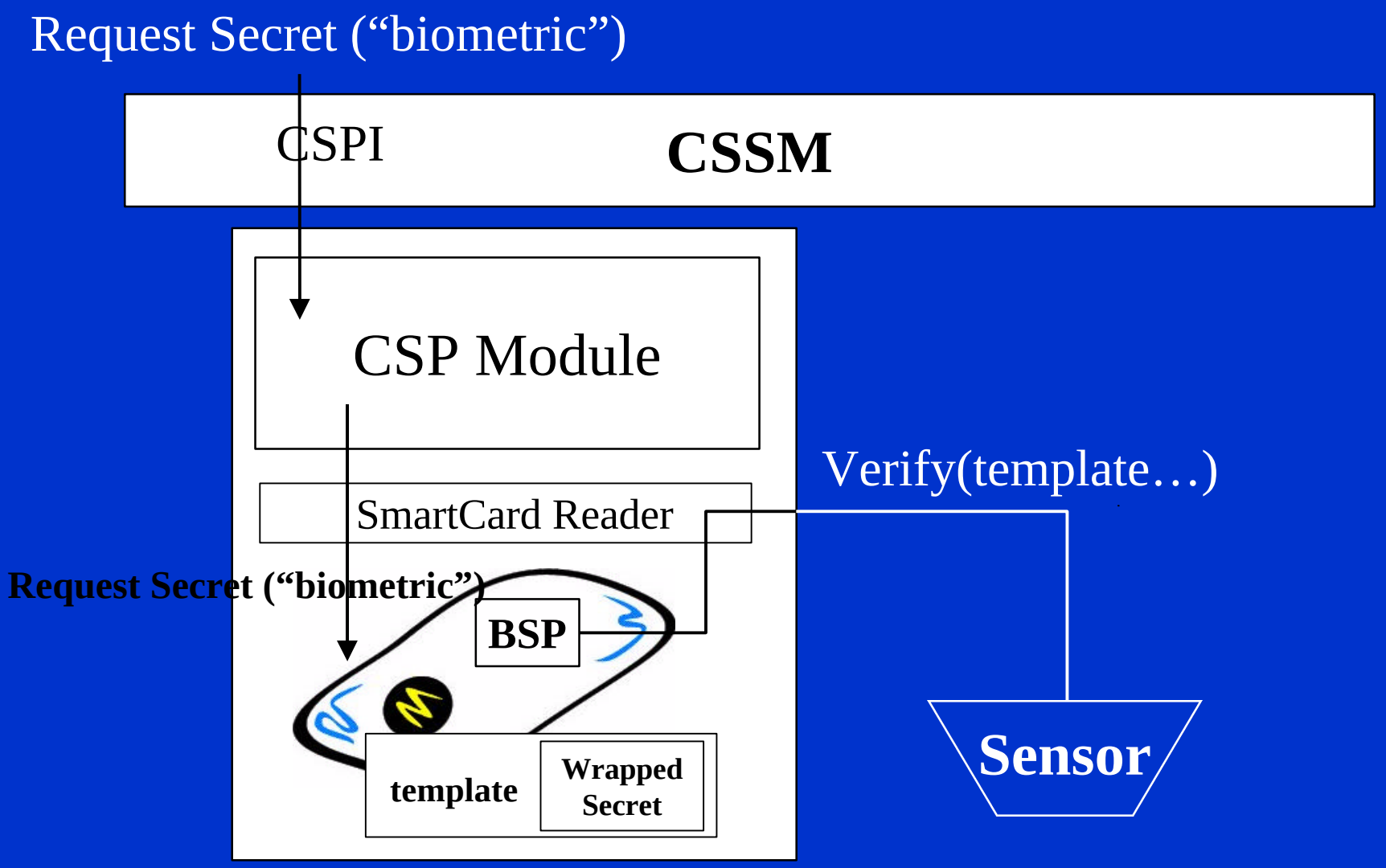
SmartCard Login



Guarding Secrets (evolutionary)



Guarding Secrets



CDSA Ubiquity