(LLD) criu-webos (5.0) (English)

1. Document Descriptions

1.1 The Purpose of Document

The purpose of this document is to describe the architecture design output of criu-webos, a component of the webOS TV 5.0 product.

Therefore, this document has been prepared for stakeholders involved in the development of criu-webos.

1.2 The Scope of Document

The scope of this document is the stakeholder, architectural driver, system context, and architecture design necessary to understand the architecture of criu-webos in webOS TV 5.0.

1.3 The Organization of Document

This document consists of six chapters as follows.

1.4 Terminology and Definitions

Definitions of terms and abbreviations used in this document are as follows.

Terms

Definition	Description
NA	

2. Project Overview

2.1 The Stakeholders

Role	Stakeholder
Project Leader	황인규 inkyu.hwang
webOS TV Architect	임창욱 changwook.im

3. Architectural Drivers

In this chapter, functional requirements for criu-webos from High-Level Functional Requirements, and architectural drivers composed of quality attributes and business-/technical-constraints are described.

3.1 Constraints

This section describes the restrictions on the design of criu-webos.

3.1.1 Technical Constraints

Restrictions on the technical side are shown in [Table 3-1].

[Table 3-1] Technical Constraints

ID	Constraint	Description
TC1	There are resources that cannot be dumped.	▪ unix domain socket ▪ system v shared memory ▪ block, character device ▪ emmc ReadWrite data
TC2	In order to apply criu to the app, it is necessary to prepare in advance.	The apps to which criu is currently applied are as follows. ▪ inputcommon ▪ livemenu ▪ home

*TC: Techincal Constraint

3.2 Functional Requirements

This section lists the functional requirements of criu-webos and additionally describes matters that require detailed examples.

3.2.1 Functional Requirements List

Functional requirements are as [Table 3-2].

[Table 3-2] Functional requirements

Group	ID	Function	Test Scenario	difficulty	importance
	FR1	When the TV performs snapshot making, criu dump is performed for the livetv app.	TVDEVTC-15716 - Getting issue details... STATUS	upper	upper
	FR2	When the TV performs snapshot making, criu dump is performed for the home app.	TVDEVTC-15718 - Getting issue details... STATUS	upper	upper
	FR3	After the TV performs snapshot making, run livemenu to proceed with criu dump.	TVDEVTC-15720 - Getting issue details... STATUS	upper	upper
	FR4	The images dumped to FR1, FR2, and FR3 above are saved in both tmpfs and emmc.	TVDEVTC-15720 - Getting issue details... STATUS	upper	upper
	FR5	For an app that has been dumped, if it is freshly launched, it is executed through the criu restore process.	TVDEVTC-15730 - Getting issue details... STATUS	upper	upper
	FR6	When the criu app to be preloaded is not on the running list, the app can be preloaded to launch.	TVDEVTC-1990 - Getting issue details... STATUS	upper	upper
	FR7	When there is a preloaded criu app in the running list, you can close the preloaded app with closeByAppId of sam.	TVDEVTC-1994 - Getting issue details... STATUS	upper	upper
	FR8	After the preloaded criu app is terminated, the corresponding app can be launched.	TVDEVTC-1996 - Getting issue details... STATUS	upper	upper
	FR9	When the dumped image does not exist, the preload launch request for the app fails.	TVDEVTC-1998 - Getting issue details... STATUS	upper	upper
	FR10	If the image dumped to ram is deleted, the dump image is copied to ram when the app is executed.	TVDEVTC-1998 - Getting issue details... STATUS	upper	upper
	FR11	If the image dumped in emmc is deleted, the dump image is copied to emmc when the app is executed.	TVDEVTC-2001 - Getting issue details... STATUS	upper	upper

	FR12	If the image dumped to emmc is broken, the dump starts anew when the app is executed, and then restore works normally.	TVDEVTC-2002 - Getting issue details... STATUS	upper	upper
	FR13	If a preloaded criu app exists, if you request the preload launch of the app again, the launch fails.	TVDEVTC-2002 - Getting issue details... STATUS	upper	upper
	FR14	If the dump image is marked as invalid, when the app is launched, a new dump is performed.	TVDEVTC-2002 - Getting issue details... STATUS	upper	upper

3.3 Quality Attribute

This section describes the quality attributes (quality factors), which are non-functional requirements that must be satisfied in criu-webos.

3.3.1 Quality Attributes

Quality attributes are shown in **[Table 3-4]**.

[Table 3-4] Quality Attributes

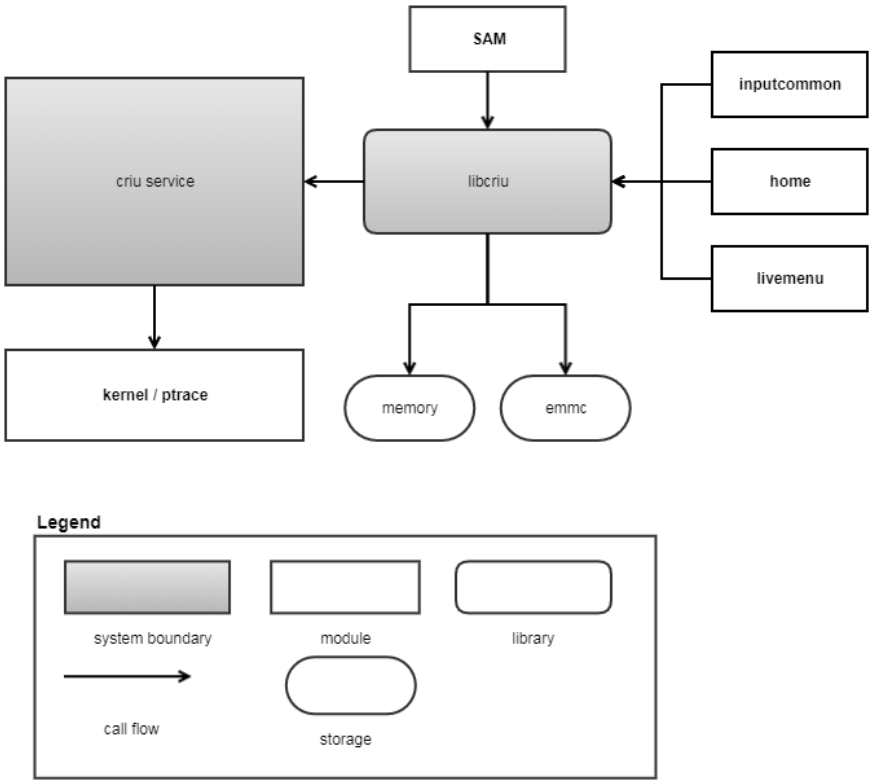
ID	Attribute	Scenario	difficulty	importance
QA1	Performance	When dump / restore, it must be completed within the time below. ▪ dump : 7 sec ▪ restore : 3 sec	upper	upper
QA2	ease of modification	The code should be divided so that it can be easily modified by separating opensource code and webos code.	middle	middle
QA3	stability	Even if criu dump / restore fails, app launch should run normally.	upper	upper

4. System Context & Interaction Model

This section describes the system context of criu-webos. External entities are identified through this system context and the system boundary is clarified through this.

The context diagram is expressed through **[Figure 4-1]**, and the responsibilities of entities and relationships constituting this system context are described in detail in **[Table 4-1]** and **[Table 4-2]** below.

[Figure 4-1] System Context Diagram



[Table 4-1] Entity Responsibility Catalog for System Context

Associated Drawings: [그림 4-1]	**Perspective : Dynamic**
Entity	**Responsibility**
criu service	Execute the requested dump
kernel / ptrace	When dumping, control the dumpee process through ptrace and provide a system call to acquire resource information
libcriu	Linked to sam and each criu app ▪ sam : When running the criu app, after determining whether to restore or not, proceed with the corresponding operation ▪ criu app : Request dump to criu service
sam	When the app to be executed is criu app, restore request to libcriu
inputcommon	Dump request to libcriu to proceed with criu dump
home	Dump request to libcriu to proceed with criu dump
livemenu	Dump request to libcriu to proceed with criu dump
memory	To improve performance during criu dump, the dump image is saved to tmpfs.
emmc	Save to emmc to keep dump image even when TV is turned off

[Table 4-2] Relationship Responsibility Catalog for System Context

Associated-Drawings: [Figure 4-1]	**Perspective : Dynamic**
Relationships	**Responsibility**
criu service → kernel	1. When dumping, adjust the dumpee process through ptrace, acquire and save the necessary resources 2. When restoring, clone the process and restore the saved resources
libcriu → criu service	If dump is requested to criu service during dump, the process is stopped through ptrace in criu service, and the dump operation proceeds.
sam → libcriu	1. Check if criu dump image exists 2. Restore in progress when dump image or zygote process exists
inputcommon / home / livemenu → libcriu	1. inputcommon / home / livemenu is executed to request criu dump during initialization 2. Even in the case of restore, it starts at the point where criu dump is returned.
libcriu → memory / emmc	1. Save dump images to tmpfs and emmc 2. When restoring, use the image stored in tmpfs or emmc

5. Architecture Design

This chapter describes the architecture design for the system. The described contents are expressed by decomposition of the system in several stages in a top-down manner. And the responsibility for the components identified by the decomposition of the system, the responsibility for the relationship between each element, and the quality attributes related to the decomposition and the rationale for the decomposition are described.

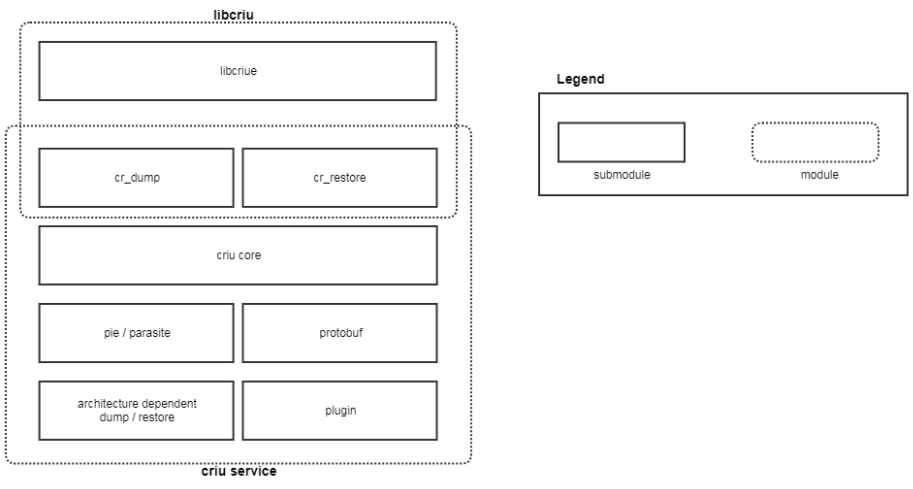
5.1 Static Perspectives

This section describes how the functions necessary for this system are decomposed based on what quality factors by developing the structure of criu-webos from a static perspective to Top-Down.

5.1.1 First Decomposition Diagram

criu-webos is first decomposed with the structure shown in [Figure 5-1] below.

[Figure 5-1] First Decomposition Diagram



Responsibilities and rationales for architectural layers expressed in the above [Figure 5-1] decomposition are as follows [Table 5-1].

[Table 5-1] Element Responsibility Catalog for First Decomposition

Associated Drawings: Figure 5-1	**Perspective : Static**
Element	**Responsibility**
libcriue	1. As a wrapping layer for using criu, self dumping, restore, and API that can check various setting values for operating criu are provided. 2. It provides callback interfaces such as pre checkpoint and post restore for application application.
cr_dump	interface module to run the requested criu dump
cr_restore	interface module for executing the requested criu restore
criu core	Module handling core functions for criu functions ▪ Get task information ▪ Obtain socket information ▪ Acquire memory information ▪ Acquire other resource information
pie / parasite	parasite binary module to be used for dump / restore ▪ The parasite code is planted and executed in the dumpee process to proceed with the dump / restore operation. ▪ In dump, the resources of the process are extracted, and in restore, the saved restores are restored.
protobuf	Use the structure module to store resources, google's protobuf
architecture dependent dump / restore	A collection of architecture-dependent C code and assembler code.
plug in	A plug-in module for exception handling during dump. ▪ fd : A code group that identifies fd that requires special processing during dump and handles it ▪ vma: A code group that identifies virtual memory areas that require special processing during dump and handles them ▪ shared memory: A code group that identifies and processes shared memory that requires special processing during dump

Responsibilities and rationales for architectural elements included in each layer are shown in [Table 5-2] below.

[Table 5-2] Rationale Description for First Decomposition

Associated Drawings: [Figure 5-1] Associated Quality Attribute:	**Rationale**
QA2(Stablility)	Separate the criu opensource code and libcriue, and the webOS application part uses only libcriue, so the code is divided for easy modification.
QA3(Stablility)	When app launch through criu fails, code for alternative operation is placed in libcriue, and the stability of webOS operation is improved.

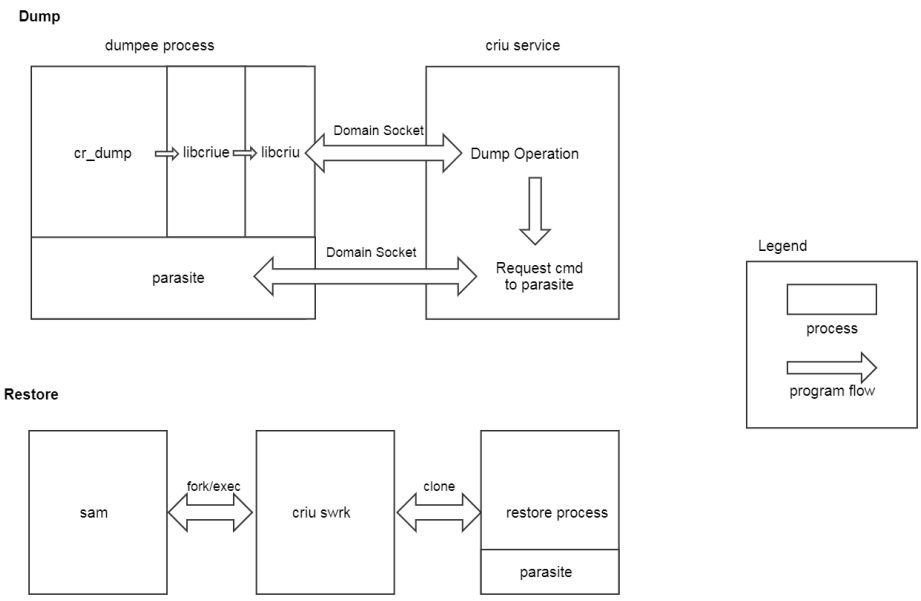
5.2 Dynamic Perspectives

In this section, the structure of criu-webos viewed from a dynamic perspective is developed as Top-Down, and it describes how the functions necessary for this system are related based on which quality factors.

5.2.1 First Decomposition Diagram

[Figure 5-2] below shows the dump / restore sequence of criu-webos.

[Figure 5-2] Second Decomposition Diagram



[Table 5-3] Element Responsibility Catalog for First Decomposition

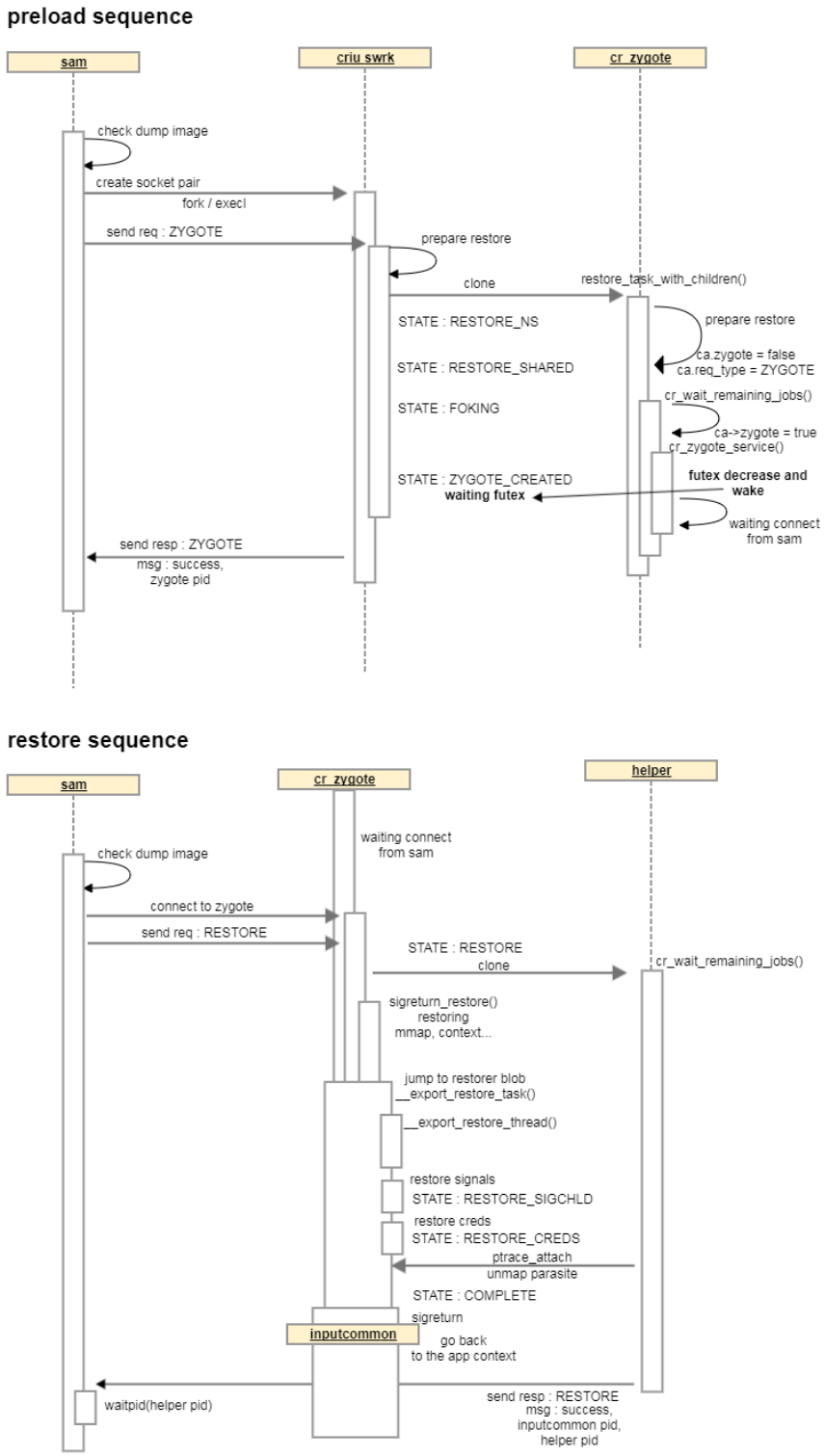
Associated Drawings: 그림 5-2	Perspective : dynamic
Element	**Responsibility**
dumpee process	When the corresponding app is executed, it requests to dump itself through the cr_dump call.
cr_dump	Check the dump condition and request dump.
libcriue	After performing various pre-works according to the webOS environment, request a dump from libcriu. ▪ save logger information ▪ Storage information to be dumped ▪ other information
libcriu	It connects to the criu service and requests to dump itself.
criu service	After saving information that can be obtained from outside the dumpee process for dump, and injecting parasite code to the dumpee process, internal information is saved Finish dump.
parasite	For dump / restore, obtain and save the information of your own process, or redeploy the saved information to your own process.
sam	When an app launch request comes, it checks whether there is a dumped image, and when there is an image, it requests restore.
criu swrk	After forking from sam, it is converted to a process that performs restore operation, and then the saved resources are relocated to its own process. When the restore process is completed, the process is terminated.
restore process	A criu swrk with certain resources relocated is cloned and created, and the remaining resources after cloning are relocated to their own process to restore the original dump condition.

5.2.2 Second Decomposition Diagram

If the dump / restore sequence of criu-webos is second decomposed into a sequence diagram, [Figure 5-3] and [Figure 5-4] are shown below.

[Figure 5-3] shows the restoration process in preload launch and preloaded app, and [Figure 5-4] shows the restoration process when there is no dump and preloaded app.

[Figure 5-3] Second Decomposition Diagram



dump / restore sequence

[Figure 5-4] Second Decomposition Diagram

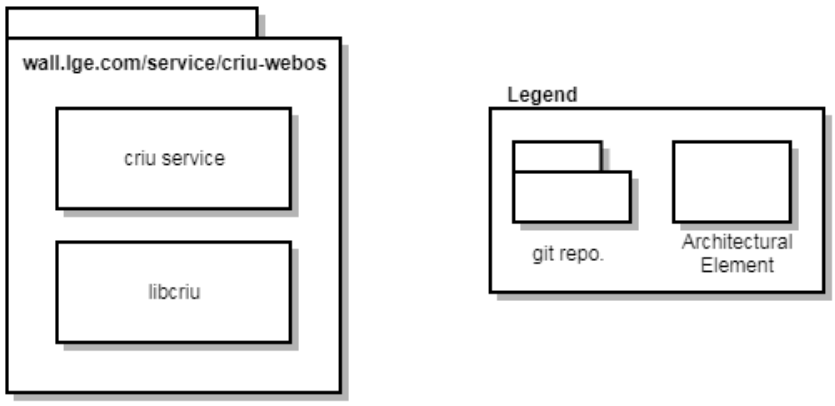
5.3 Physical Perspectives

When looking at criu-webos from a physical perspective, it describes the repository of the source code.

5.3.1 Decomposition Diagram of Source Repositories

The source code of criu-webos is designed as a Git repository structure as shown in [Figure 5-1].

[Figure 5-1] A Decomposed Source Repository Map of criu-webos



6. Reference

If there are related materials for reference, please describe them in the table below or attach related materials.

no	Item	Attach related documents
6.1	API Document	*NA*
6.2	Database	When using a database, link to the relevant jira or collab page where the necessary data information is defined.
6.3	Patent issues	Check whether the patent evasion method is reflected in the design and link to related materials.
6.4	Other References	09. CRIU를 통한 App/Service Launching 시간 개선 00. CRIU 03. CRIU