Long Term Support Initiative

- Project Overview -

March 7, 2012

CE Workgroup
The Linux Foundation

THE LINUX FOUNDATION

Preface

The CE workgroup (CEWG) in the Linux Foundation has conducted intensive discussions to identify root causes of inefficiency in product development of Linux based system, and less contributions to upstream from industry engineers. CEWG now proposes a mechanism to solve these issues, and contribute to the further evolution of Linux.

This document describes an overview of the "Long Term Support Initiative for the Industry (LTSI)" project, which is expected to solve the above issues. It is divided into the following six sections, executive summary and a set of appendices:

'Executive Summary' consolidates the industry pain points and solutions provided by the LTSI project. All readers are advised to read this section.

Section 1: *'Introduction'* provides the background of this project, and a brief overview of the LTSI project.

Section 2: *'LTSI Project Overview'* describes the LTSI team structure, operation flows, and relationship with the community's long-term-stable tree. Expected patch-flows from industry engineers to upstream are described as well.

Section 3: *'LTSI Resource Definitions'* describes the role and responsibilities of each member in the LTSI project.

Section 4: *'LTSI Release and Maintenance Operation'* describes the basic process to release an LTSI kernel tree, and maintain it. The level of participation to the LTSI project is also described.

Section 5: *'LTSI Operation Rules'* describes the basic rules to accept patches from industry engineers, semiconductor vendors, and other contributors, and describes the rules to back-port features from upstream into the LTSI tree.

Section 6: *'LTSI Value Propositions'* describes value propositions for ecosystem partners as well as the consumer electronics industry.

Section 7: *'Summary'* provides the summary of a project proposal, status, and steps to start actual operations.

A set of appendices provide additional information.

Table of Contents

Executive Summary

The LTSI project is an ecosystem-wide collaborative project to create and maintain a common Linux base for the use in a variety of products, to be hosted by the Linux Foundation. The project creates and maintains the long-term support industry tree, the LTSI tree, which is expected to be stable in quality for the typical life-time of a product, i.e., 2-3 years. The LTSI tree is created based on the community's long-term-stable tree, incorporates some valuable features from ecosystem partners, and is maintained by back-porting bug fixes and some valuable features from the latest upstream kernel version. The use of the LTSI tree significantly reduces the burdens on industry engineers who need to enable utilize versions of Linux in a variety of products, and maintain those individually.

The LTSI project also has a mechanism to accelerate contributions from industry engineers in two different ways. The first path is to help industry engineers to send patches upstream by preparing the industry staging tree where innovative but possibly incomplete features can be posted. Then, the LTSI team and other industry engineers can discuss these improvements. Upstream maintainers will be involved appropriately. The second path is to accept fixes and features into the LTSI tree for further brush-up towards upstream by the LTSI team. This allows industry engineers to easily send fixes and features, since they should be using the same version of Linux as the LTSI tree. And, as the same rules described in the community's Documentation/SubmittingPatches are applied for sending patches to the LTSI tree, industry engineers will become knowledgeable in the community process which should eventually lead them to contribute directly to upstream. In parallel, the LTSI team works with upstream engineers to upstream those patches.

The LTSI project can not be done by the industry alone. It requires a tight linkage with the community's long-term-stable tree maintainer to select a base version of Linux for wider use in the industry, and close communications with upstream maintainers to incorporate fixes and innovations from industry engineers. The Linux Foundation is the only organization which can bridge the industry and the community for the evolution of Linux and the use of Linux in the business, and believes the LTSI project contributes to both.

1 Introduction

1.1 Background

The consumer electronics industry has been rapidly growing using Linux as a base for many products, and incorporating many new features in the operating system. But, contributions by the industry to upstream are not outstanding compared to innovations that the industry have been incorporating.

One of reasons identified for this disparity is the fragmented use of Linux and the version gap caused by it. In this industry, some implementations of Linux come from semiconductor vendors as Board Support Packages (BSP), some are based on internal Linux trees which were "forked" a long time before, and some come with Android. There are version gaps, and it is hard to share knowledge across projects. And, the fragmented use of Linux and time-pressure makes it difficult for embedded system engineers to use an "upstream first" policy, but rather forces them to work locally on their product releases. The forked private tree is far from the upstream version due to the version gap and numerous private fixes that makes it difficult to even test with the latest upstream.

The other reason is that, traditionally, the industry tends to keep modifications and enhancements in secret within a company, since those are considered as valuable "know-how" and differentiation. Few engineers are assigned to work with upstream in many companies.

But, with increasing use of Linux in consumer electronics products, the fragmented use of Linux and continuously applying private fixes to multiple versions of the kernel are becoming major pain points in the industry. Now, the industry is longing for long-term support of Linux as a common base for all products which also accelerates upstreaming of private fixes and features by eliminating duplicate effort to work on different versions. And, a long-term support version of Linux can be a common base for the entire ecosystem including semi-conductor venders, set-vendors, software component vendors, distributors, and hopefully Google Android and Tizen to solve the fragmentation issue.

On the other hand, requirements for long-term support in the consumer electronics industry and in the enterprise arena are different due to the product lifecycle and composition of the ecosystems. For example, the consumer electronics industry does not require 10-years long-term support, but do require multiple 2-3 years long-term support versions to fit to the product lifecycle. As a result, we have decided to kick off "Long Term Support Initiative", and propose "Long Term Support of Linux for the Industry", in short LTS for the Industry or LTSI, as an industry wide activity including ecosystem partners to fill the gaps.

1.2 LTSI - What it is?

The consumer electronics industry has slightly different requirements on long-term-stable tree in addition to the duration of support. By definition, fatal bug fixes and security fixes will be incorporated into a stable tree. (See Appendix A2) On the other hand, the industry needs back-porting of features for emerging device support, performance improvement, better power management capability, and so on. And, features to be back-ported may vary by industry (e.g., consumer electronics, aerospace and defense, automotive, control system, and others.)

The LTSI project is an industry-wide collaborative activity to minimize fragmentation, and accelerate further innovation of Linux by incorporating fixes and innovations from embedded system engineers. The LTSI tree is expected to be the usable as a base for the majority of embedded systems, as well as the base for ecosystem players (e.g., semiconductor vendors, set-vendors, software component vendors, distributors, and system/application framework providers such as Google Android and Tizen).

The LTSI tree is maintained by the LTSI Back-Port (BP) team and ecosystem partners. Basically, base trees like (MM, PM, RT and Security) are maintained by LTSI BP engineers, and device/architecture trees are expected to be maintained by ecosystem partners like processor/SoC vendors. The basic rules are described in section 5.1, but details will be discussed with processor/SoC vendors.

The LTSI project accepts important features from contributors, BSP authors and industry engineers which are not incorporated into upstream in some reasons, but are still beneficial for the industry. This is not intended to create another version of Linux, but is designed as a placeholder for several important features which may require a certain amount of time to be incorporated into upstream. The LTSI team will keep working with upstream maintainers to find the best way to upstream features.

Basically, a new LTSI tree will be released every year based on requirements from the industry. And, the LTSI tree will be maintained for two years, and, at most two LTSI trees will be maintained in parallel at one time. (Note that the release schedule may vary based on requirements from the industry such as the version and duration. Details will be discussed with ecosystem players.)

1.3 LTSI - What it is NOT.

The LTSI project is NOT intended to discourage the use of upstream kernel. By reducing the fragmented use of Linux, CEWG expects industry engineers to find more time to work with upstream, rather than working on several versions of Linux locally.

The LTSI project is NOT intended to discourage industry engineers to contribute to upstream. CEWG will conduct regular back-porting work to the LTSI tree from the upstream kernel. Bug fixes and some key important functional enhancements that industry engineers have contributed to upstream will be ported into the LTSI tree. There will be no need to apply such changes to their private trees any more. This significantly reduces local work by industry engineers, and motivates them to work with upstream. And, by reducing the gap between the LTSI tree and the upstream kernel by way of feature back-porting, CEWG expects industry engineers to easily test with the latest version of the upstream kernel, and work with upstream engineers to find better solutions.

Creating the LTSI tree does NOT mean the embedded industry wants to stay with an old version of Linux. Similar to the Enterprise, the embedded industry may want to use a single version of Linux for the life of a particular product. The evolutions of SoCs and devices in the embedded industry are quite fast comparing to the Enterprise. The embedded industry can not stay in an old version of Linux for long years, since several new features are supported in upstream. The industry needs to moves to the latest upstream periodically. But, even in a relatively short maintenance period, the evolutions of SoCs and devices are fact enough. Feature back-porting to the LTSI tree is necessary to keep the value of the LTSI tree. The industry is ready to pick up the latest version, but expects it to be stable while a product is in the market, i.e., for 2-3 years.

2 LTSI Project Overview

2.1 LTSI Project Overview

The LTSI project consists of three major activities as follows.

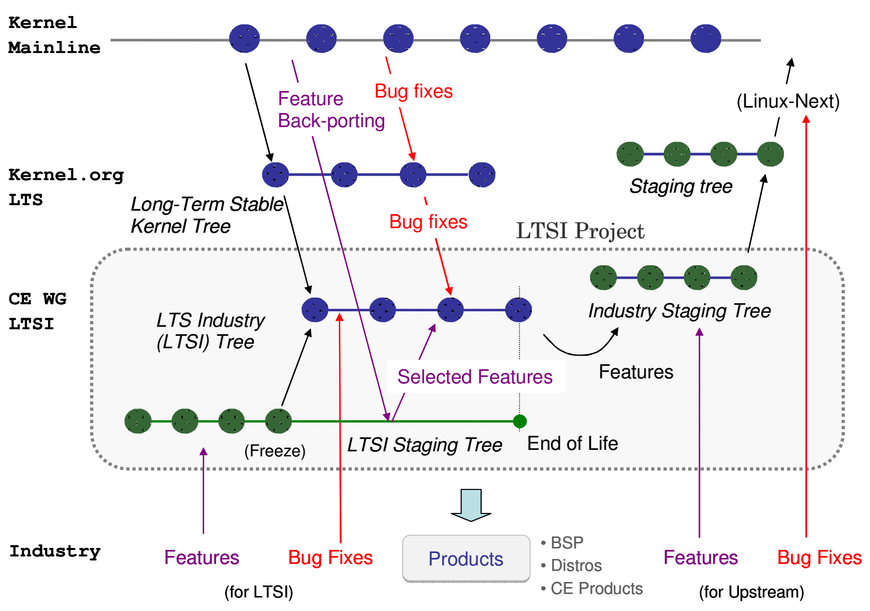


Figure 1 LTSI Project Overview

1. Create and maintain a long-term stable tree in kernel.org.
2. Create and maintain the LTS industry tree (LTSI tree) and the LTSI staging tree associated with it.
3. Create and maintain an industry staging tree, evaluate and filter features, and promote them to upstream through the community's staging tree.

The first activity is basically owned by the Linux community, but the LTSI team wants to work closely with the community to identify an appropriate version to be a long-term stable tree, and decide its duration to maintain. From the industry view point, a long-term stable tree is expected to be created once per year with two-year support. But, no more than two long-term stable kernels are expected to be maintained at any one time due to limitations in the resources needed to perform this activity.

The second activity is to maintain the LTSI tree and sustain its value for the industry. New features in upstream are back-ported into the LTSI tree, and additional features which are not in upstream may be incorporated into the LTSI tree through the LTSI staging tree. Flows to accept non-upstream features into the LTSI tree are described in section 2.3, and rules to accept them are described in section 5.2. This industry tree may be maintained slightly longer than the long-term stable tree in the community, but it can not be much longer after the termination of support to the community's long-term stable tree. Two years for both are an initial assumption, but, further discussion among ecosystem partners will be required.

The third activity is to promote upstreaming of fixes and features from the industry. The industry staging tree is prepared for testing and consolidating features from different companies. An industry staging tree maintainer evaluates and filters features from the industry, and upstreams them through the community's staging tree. In addition, some consulting services will be offered to industry engineers to help them to work with upstream maintainers.

Detail work flows are described in section 2.3.

2.1.1 LTS Industry (LTSI) Tree

The LTS Industry Tree, the LTSI tree in short, is a primary resource and outcome of this project. It is created based on the community's long-term stable tree, roughly once per year, and maintained by the LTSI team for 2 years as an initial assumption It includes bug fixes ported from a long-term stable tree, and new features directly ported from the latest upstream. It also accepts inclusions of some features which are not included in upstream for some reason, but which are still deemed quite beneficial to the industry. Note that the LTSI team does not promote a policy of including features directly into the LTSI tree first, but still recommends an "Upstream First" policy to the industry. The rule to incorporate non-upstreamed features are described in section 5.2. This tree is expected to be usable as a common base for many products from the industry.

The LTSI tree will be updated on a regular basis to incorporate bug-fixes and important security patches from a long-term stable Kernel tree in the community. New features will be back-ported to the LTSI staging tree for cherry-picking by a user, and some selected features will be merged into the LTSI tree. The selection criteria will be defined later, but all features to be merged should not break anything and keep the compatibility through the life of the LTSI tree. At most two LTSI trees will be maintained at one time.

2.1.2 LTSI Staging Tree

The LTSI staging tree is created to hold features which are not included in upstream, but expected to be included in the LTSI tree due to importance to the industry. The role of tree is similar to that of the linux-staging tree, but, it accepts not only drivers and file systems but also other features which are beneficial for the industry. As the LTSI project does not have the linux-next like tree, mature features will be directly pulled from here by the decision of the LTSI chief maintainer with the guidance from the CEWG Architecture Group at the creation of the LTSI tree.

2.1.3 Industry Staging Tree

The industry staging tree is created to filter and coordinate features from the industry to be smoothly integrated into upstream through the community's staging tree. An Industry Staging Tree maintainer moves mature features into the community's stating tree for discussions with broader community engineers. The reason why a separate staging tree is prepared is that the majority of industry engineers are not mature enough to discuss with upstream maintainers today, and need pre-discussions, re-implementation and some preparations including well-described text for features before posting to the community's tree. If industry engineers become mature, then, this tree will be discontinued.

The base version of the industry staging tree is same as that of the community's staging tree, and its life is exactly same as that of the community's staging tree.

2.1.4 Long-Term Stable Tree in the Community

A long-term stable kernel tree is created by a community stable tree maintainer basically once per year and maintained for approximately two years.

The Industry Advisory Board (IAB) is a newly proposed organization in the Linux Foundation that discusses and advises which version of upstream kernel is expected to be a long-term stable tree from an industry view point. It works with a community stable tree maintainer to decide the version to be a long-term stable tree. The IAB will consists of representatives from several industries including final product makers, semiconductor manufacturers, distributors, and application platform providers

2.2 LTSI Project Team Structure

The LTSI project consists of two key tasks to maintain internal resources. One is to create and maintain the LTSI tree which is a common base for product development in the industry, and the other is to promote upstreaming by maintaining the industry stating tree which functions as a stepping stone to the community's staging tree. Both teams are managed by a project manager and technically led by the LTSI chief maintainer who has good capabilities to maintain long-term-stable trees and staging trees. The overall team structure is shown in Figure 2.

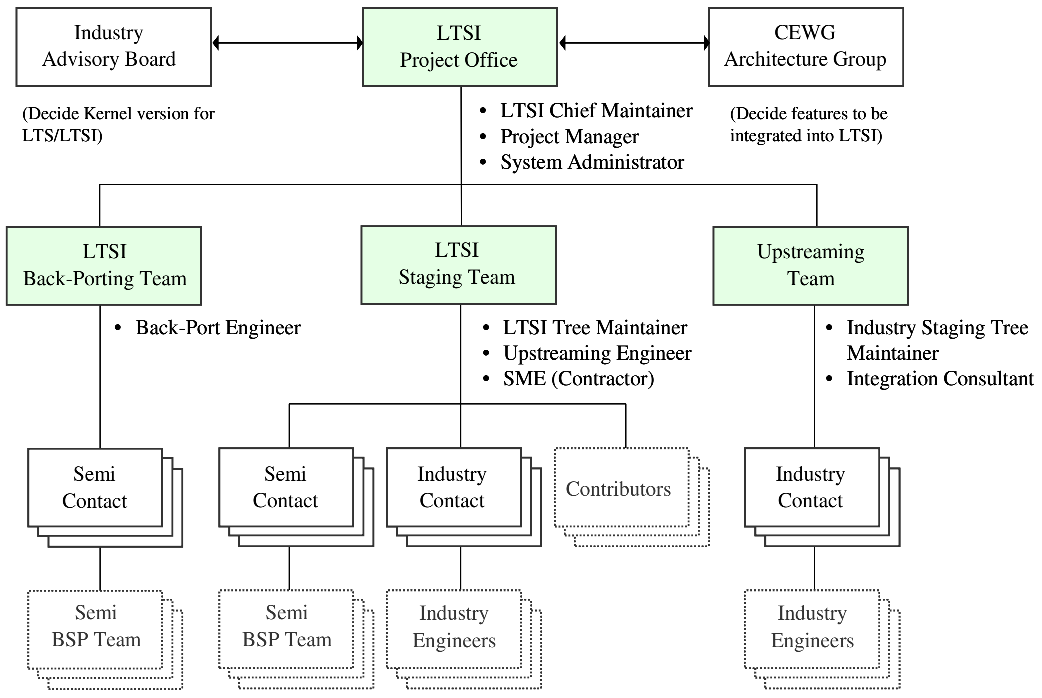


Figure 2 LTSI Team Structure

The entire LTSI team is led by the LTSI chief maintainer who has a responsibility to sustain the quality of the LTSI tree, carefully decides which fixes and features should be incorporated into the LTSI tree with the guidance from the CEWG Architecture Board. This team has three sub-teams as follows.

1. LTSI Back-Port (BP) Team: This team creates the LTSI tree based on a long-term-stable tree in the community, maintains it with incorporating bug fixes and security fixes from a long-term-stable tree, and back-ports valuable features from the upstream tree to sustain the value of the LTSI tree.
2. LTSI Staging Team: This team works with multiple groups in the industry to incorporate valuable features into the LTSI staging tree, selects appropriate features with guidance from the LTSI chief maintainer, and merges them into the LTSI tree. Basically, features that this team incorporates are important and useful features which were rejected by upstream maintainers for some reason, but which are still deemed beneficial to the industry. (An example is RT-preempt.) The rules to accept features from the industry are described in section 5.2. This team has another important role. In order to accelerate participation and contributions of industry engineers to upstream, LTSI staging team also accepts fixes and features from industry engineers into the LTSI staging tree. Those fixes and features are evaluated and examined by the team, and upstreamed by either an upstreaming engineer in the

LTSI staging team or a contractor who is a subject matter expert (SME) in a particular area that a fix or feature is applied.

3. Upstreaming Team: The upstreaming team maintains the industry staging tree that includes new features which are expected to be incorporated into upstream through the community's staging tree, encourages industry engineers to send their fixes and new innovations, and helps industry engineers to upstream their patches. This team also accepts patches from LTSI staging team which were originally developed by industry engineers for the version of LTSI, and refactored for upstream.

2.3 LTSI Operations Overview

Figure 3 shows an overview of LTSI operation flows across multiple resources. The LTSI project driven by CEWG in the Linux Foundation owns the LTSI tree (LTS Industry tree), the Industry staging tree, and the LTSI staging tree associated to the LTSI tree.

1. A community engineer creates and maintains a long-term-stable tree in the community, then, back-ports critical bug fixes and security fixes periodically.

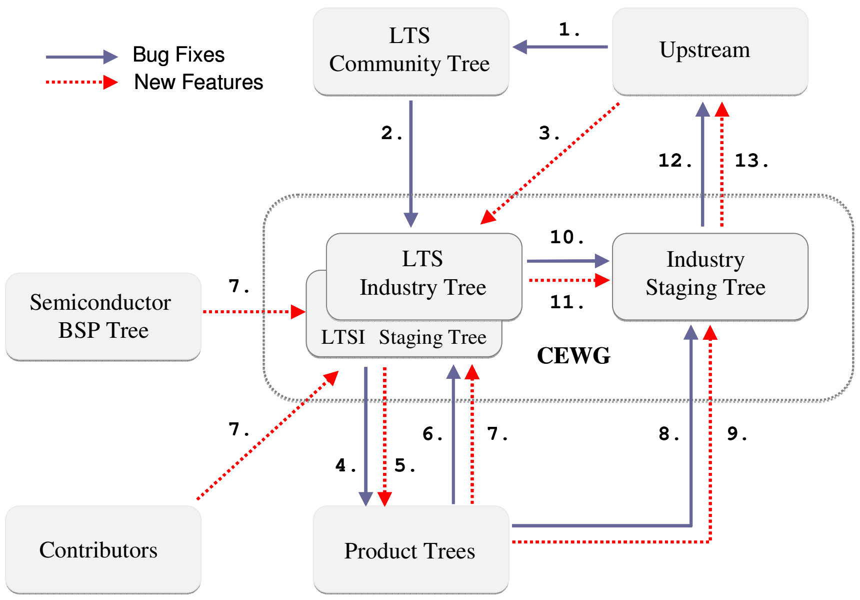


Figure 1 LTSI Operation Flows

2. The LTSI project team creates and maintains the LTSI tree. LTS BP engineers pick up fixes from a community's long-term-stable tree appropriately.
3. In addition to 2, LTS BP engineers back port some valuable features from the upstream. In parallel, the LTS chief maintainer accepts selected feature-back-porting by semiconductor engineers.
4. Industry engineers use the LTSI tree as the base for their product development, and periodically pull bug-fixes.
5. Industry engineers also pick up some new features from the LTSI tree, if required.
6. Industry engineers send bug fixes to the LTSI staging team. A fix can be one of two types. It can be a fix to a feature which is not in upstream. Or it can be a fix, intended to eventually be sent to upstream, but which is sent to the LTSI tree at this time due to resource problems on industry engineers' side. The reason why the LTSI staging team accepts the latter type of a fix is described in section 2.4.
7. Industry engineers also send features to the LTSI staging team. Similar to the flow 6, a feature can be one of two types. It can be a feature which was not accepted by upstream maintainers for some reason, but is still considered beneficial for the industry. Or it can be a feature, intended eventually to be sent upstream, but which is sent to the LTSI tree at this time due to resource problems on industry engineers' side. The reason why LTSI staging team accepts the latter type of a feature is described in section 2.4.
8. Industry engineers work with an industry staging tree maintainer and an integration consultant to discuss the way to incorporate bug fixes into upstream with involving a community maintainer. Of course, mature industry programmers can send a patch directly to upstream working with upstream maintainers.
9. Industry engineers send features to the industry staging tree. Features will be evaluated and filtered by industry staging tree maintainers, and, eventually, posted to the community's staging tree to be pulled to upstream. Industry engineers can consult with an integration consultant in the upstreaming team to find the best way to incorporate features with upstream maintainers.

10. LTSI upstreaming engineers continuously work with an industry staging tree maintainer and upstream maintainer to upstream fixes.
11. LTSI upstreaming engineers continuously work with an industry staging tree maintainer and upstream maintainer to upstream valuable features through the industry staging tree.
12. An industry staging tree maintainer and an integration consultant help industry engineers to send bug fixes to upstream.
13. An industry staging tree maintainer posts appropriate features into the community's staging tree, and follows up with upstream engineers to be integrated into upstream through Linux-Next.

2.4 Promotion of Upstreaming

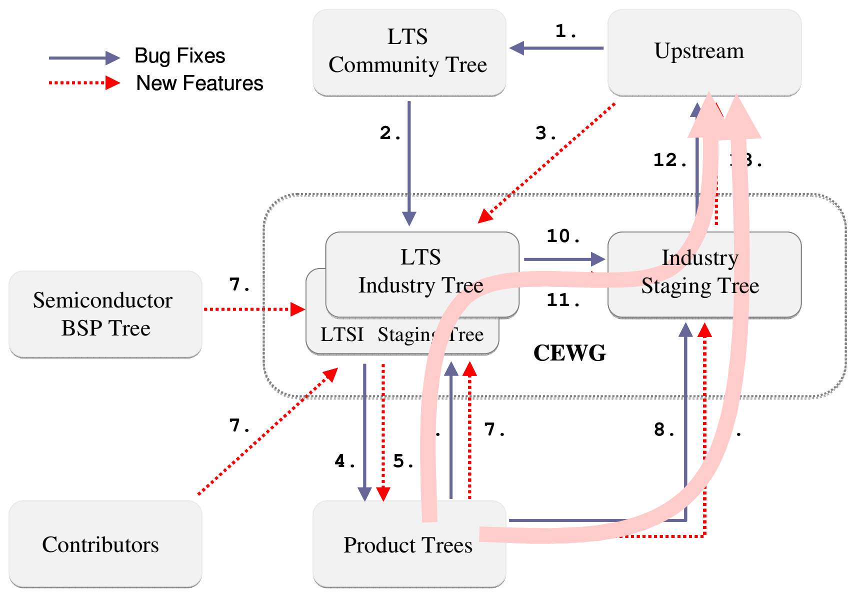


Figure 2 Paths to Upstream from the industry

As seen in Figure 4, there are two paths to upstream from industry engineers. Eventually, as the industry matures, both paths may not be necessary. Instead, industry engineers may work directly with upstream maintainers to send patches to upstream as described in Appendix A1.

But, the fact is that there are few engineers who can directly work with upstream maintainers. In order to cultivate industry engineers and encourage them to work with upstream maintainers, the industry staging tree will be created. An industry staging tree maintainer and integration consultant will work with an industry contact in each company, pull patches, evaluate and filter them, and promote to upstream.

And, one of the major reasons that industry engineers can not work closely with upstream maintainers is version gap. Although the LTSI project reduces version fragmentation by providing a common base for a variety of products, there is still a version gap between LTSI and the latest upstream kernel. Industry engineers are usually working on a product tree, and do not have time to re-implement it for the latest version. Or, they even do not have the environment to work with the latest upstream tree. This is a long-lasting problem which has not been solved for many years. To help industry engineers become aware of community process without imposing additional burdens on them, CEWG decide to accept fixes and features into the LTSI tree or the LTSI staging tree using the same signed-off-by process that is used in mainline. It should be easier for industry engineers to contribute to the LTSI staging tree, since they should be using the same version of Linux, i.e, LTSI. Then, a LTS industry tree maintainer reviews a fix or feature, re-implements it on the latest upstream kernel, and sends patches to upstream.

Both paths are tactical, but are required to create a better relationship with the community as the first step.

Another aspect to promote upstreaming is to create better connections with industry engineers. Each company that participates in the LTSI project identifies an industry contact. The LTSI staging team and the upstreaming team work with industry engineers through industry contacts for promoting and helping the upstreaming of their patches.

3 LTSI Resource Definitions

In this section, the role and responsibility of each resource in the LTSI project are described.

3.1 LTSI Project Office

The LTSI project office is a management body of an entire project to provide project management, infrastructure support and technical guidance. It also works with several external resources including the press to publish outcomes of this project appropriately. The role and responsibility of each person are described below. Note that this is an initial definition, and will be updated accordingly.

LTSI Project Manager:

- LTSI Release Management
  - Creates and publishes a release plan which includes the date to create the LTSI tree, dates of regular updates, and duration to maintain it.
  - Works with the LTSI chief maintainer and publishes release notes
  - Coordinates back porting schedules with the LTSI chief maintainer
  - Coordinates back porting schedules with SoC vendors
  - Coordinates feature integration schedules with SoC vendors
  - Coordinates feature integration schedules with contributors
  - Coordinates above activities for all active LTSI trees.
- LTSI Resource Management
  - Hires appropriate resources for the LTSI project
  - Works with an LTSI system administrator to prepare, maintain, and update infrastructure based on requests from an LTSI chief maintainer and an industry staging tree
- LTSI Budget Management
  - Works with Mike Woster, COO of the Linux Foundation, to secure budget for the LTSI project including some contractor works for upstreaming
- LTSI Liaison Coordination
  - Works with partners and the industry to appoint a contact person from each company for soliciting bug fixes and new features
- LTSI Public Relations
  - Works with the press to advertise the value of the LTSI
  - Updates highlights in the LTSI Web site.
- LTSI Contract Work
  - Works on contracts to hire subject matter experts to upstream patches and features from industry engineers if required

LTSI Chief Maintainer:

- LTSI project technical coordination

- Has the responsibility to create and maintain the LTSI tree working closely with the community. Receives the guidance from the CEWG Architecture Group.
- Works with a project manager to technically coordinate LTSI BP team and LTSI Staging Team to meet a release schedule.
- Maintains and updates rules for back-porting features from upstream, integrating features from BSP authors and contributors, and accepting fixes and features to the LTSI tree from industry engineers.

- LTSI back-porting technical coordination
  - Selects features to back-port into the LTSI stating tree, and gives guidance to LTS BP engineers.
  - Works with semiconductor vendors to coordinate back-porting features from upstream.
- LTSI staging technical coordination
  - Selects features from the LTSI staging tree based on the guidance from the CEWG Architecture Group, and gives guidance to a feature integrator to incorporate selected features into the LTSI tree.

LTSI System Administrator:

- LTSI IT Infrastructure Management
  - Establishes IT infrastructure for the LTSI tree, the LTSI staging tree and the industry staging tree, and provide administrative services.
- LTSI Build
  - Has a responsibility to perform nightly builds
- LTSI Web Site Management
  - Working with the LF Web Management, creates and maintains the LTSI Web site to be usable for the industry and ecosystem partners. It should include release schedules, status, new features, the way to contribute on the LTSI, pointers to related resources, and so on.

3.2 LTSI Team

LTSI Team has three sub-teams, i.e., LTSI BP Team, LTSI Staging Team, and Upstreaming Team. All teams receive technical guidance from the LTSI Chief Maintainer, and, are managed by an LTSI project manager.

3.2.1 LTSI BP Team

LTSI BP Team has a responsibility to create and maintain the LTSI tree, and back-ports bug fixes and new features from upstream.

LTSI Back-Port Engineer

- LTSI tree creation and maintenance
  - Receives guidance from the LTSI chief maintainer, creates LTSI tree based on the community’s long-term-stable tree, and works with a feature integrator to incorporate selected features.

- Ports bug fixes from the community's long-term-stable tree, and back-ports selected features into the LTSI tree.
- Provides information about bug fixes and features to an LTSI project manager to be incorporated into a release note.

3.2.2 LTSI Staging Team

LTSI Staging Team has a responsibility to maintain the LTSI staging tree, integrates features from semiconductor vendors and contributors, and receives bug fixes and features from industry engineers to be integrated into the LTSI tree as well as upstream. The team contacts with industry engineers through an industry contact in each company for communications.

LTSI Feature Integrator

- LTSI staging tree management
  - Creates and maintains the LTSI staging tree.
  - Works with semiconductor vendors and contributors to solicit valuable features into the LTSI staging tree.
- Feature integration into the LTSI tree
  - Reviews features with the LTSI chief maintainer, decides which features should be incorporated into the LTSI tree, and integrates them. .
  - Provides information about new features to an LTSI project manager to be incorporated into a release note.

LTSI Upstreaming Engineer

- Proxy for industry engineers
  - Receives bug fixes from industry engineers, evaluates them with the LTSI chief maintainer, and incorporates it into the LTSI. Then, re-implements for upstream with upstream maintainers, and sends patches to upstream.
  - Receives new features to the LTSI staging tree from industry engineers, and evaluates them with the LTSI chief maintainer and an LTSI feature integrator. If a feature is worth to be incorporated into upstream, then, an LTSI upstreaming engineer re-implements it for the latest upstream version, consults with upstream maintainers, and sends patches to upstream through the community's staging tree (or through the industry staging tree). If a feature requires higher domain skill, then, an LTSI upstream engineer ask an LTSI project manager to hire a subject matter expert to work on upstreaming.

Contractors (Subject Matter Experts)

- Upstreaming of fixes or features
  - Is hired if fixes or features from industry engineers require domain skills to re-implement to the upstream version of Linux.
  - Reviews and re-implement fixes or features from industry engineers, works with upstream maintainers to upstream them..

3.2.3 Upstreaming Team

The upstreaming team has a responsibility to create and maintain the industry staging tree, and is a technical interface to industry engineers for upstreaming bug fixes and features. The team communicates with industry engineers through an industry contact in each company.

Industry Staging Tree Maintainer

- Industry staging tree management
  - Creates and maintains the industry staging tree
  - Reviews features in the industry staging tree, filters them, and moves them to the community's staging tree.
  - Follows up with upstream maintainers to integrate features into upstream kernel.

Integration Consultant

- Consulting on upstreaming
  - Provides technical advice to industry engineers about upstreaming, and introduces appropriate upstream maintainers or subject matter experts to help them with upstreaming.
  - Works with an LTSI upstreaming engineer to find the best way to upstream fixes and features from industry engineers which are sent for the LTSI tree.

3.3 CEWG Architecture Group

The LTSI project is driven by the LTSI team, but the CEWG Architecture Group (AG), which consists of representatives from the industry, plays a key role to decide features to be included in an LTSI release. The CEWG AG prioritizes important features to be included in an LTSI release based on requirements from the industry, and finally submits a recommendation list to the LTSI chief maintainer. The CEWG AG also works with contractors of CEWG open projects to incorporate outcomes into an LTSI tree. .

3.4 Summary of resources for the LTSI project

Table 1 shows the summary of required resources for the LTSI project. An LTSI project manager and the LTSI chief maintainer are full-time resources, but others are basically part-time resources. In the first year of the operation, one single LTSI tree is maintained. But, from the second year of the operation, multiple trees are expected to be maintained which requires additional resources for back-porting fixes and features, and for incorporating fixes and features from the LTSI staging tree. Therefore, additional resources will be required from the second year of the operation. The overall resource plan will be defined separately.

Project Resources	**Team**	**Full-time Resource**
LTSI Project Manager	LTSI Project Office	Yes
LTSI System Administrator	LTSI Project Office	Part time
LTSI Chief Maintainer	LTSI Project Office	Yes

LTSI Back-Port Engineer	LTSI BP Team	Part time
LTSI Feature Integrator	LTSI Staging Team	Part time
LTSI Upstreaming Engineer	LTSI Staging Team	Part time
Industry Staging Tree Maintainer	Upstreaming Team	Part time
Integration Consultant	Upstreaming Team	None at Kick off, then Part time
Contractor (for upstreaming)	LTSI Team/LTSI Staging Team	None at Kick off, then Part time

Table 1 Summary of resources for the LTSI project

4 LTSI Release and Maintenance Operation

4.1 LTSI release process

An LTSI kernel tree will be released roughly once per year. The process to release an LTSI kernel tree is as follows.

Step 1: Open an LTSI staging tree for soliciting valuable features for the embedded industry

Soon after a long-term stable kernel is created in the community, the LTSI team opens an LTSI staging tree associated to that, and announce its availability. The date to freeze a tree is also announced. In parallel, the CEWG AG decides initial features to be included as the base, and asks appropriate persons or companies to start a patch submission process.

Step 2: Freeze an LTSI staging tree and Select features to be included in the LTSI release

After a certain period of time, the LTSI team tentatively freezes an LTSI staging tree to start a merge process. The CEWG AG decides final features to be included based on industry requirements, and inputs to the LTSI chief maintainer. Then, the LTSI chief maintainer finally decides features to be included with additional considerations on code maturity and conflicts to existing features.

Step 3: Publish an LTSI tree, and Start the maintenance process for 2 years

After ensuring it can be booted up on selected environment, an LTSI tree is published. The LTSI team prepares a web site to share the result of testing on any boards or products so that any vendors can post the result of testing on their boards or products to share the result with the broader audience. The LTSI team re-opens an LTSI stating tree for additional back-porting of new features during the maintenance period.

4.2 LTSI maintenance process

Basically, the LTSI tree is maintained for two years after its release. For the LTSI tree, the LTSI team conducts back-porting of bug fixes and security features periodically from an associated long-term stable kernel tree in the community to make it stable for the product use. The LTSI team also applies bug fixes submitted by industry engineers who should find hidden bugs during the intensive testing for the productization.

In parallel, valuable new features are also back-ported into the LTSI Staging tree so that any users can cherry-pick required features into their product tree. Some features will be incorporated into the LTSI tree to reduce additional burden to cherry pick, but those features will be carefully reviewed and selected not to break anything for the compatibility. The selection criteria will be announced before starting the maintenance process. The LTSI staging tree will be kept active until the end of the life of an associated LTSI tree.

4.3 Participation to the LTSI Project.

The LTSI project is an open project. Anyone can use the LTSI tree, and also contribute to the LTSI project. It does not require being a CEWG member or a LF member. The LTSI mailing list is open for everyone. Everyone can join open discussions about the LTSI. However, there are some additional benefits for CE workgroup members as follows.

Category	**Benefits**
CEWG AG Voting Member	Use the LTSI tree, submit patches, and join discussions to decide features to be merged with a voting right. An industry contact can be assigned in a company for close communications with the LTSI team.
CEWG AG Member	Use the LTSI tree, submit patches, and join discussions to decide features to be merged. An industry contact can be assigned in a company for close communications with the LTSI team.
CEWG Member (non-AG member)	Use the LTSI tree, submit patches, and receive periodical newsletters.
Non Member	Use the LTSI tree and submit patches

Table 2 LTSI Participation Level

5 LTSI Operation Rules

5.1 Feature Back-porting rules

These rules are applied to feature back-porting from upstream into the LTSI tree. If bug fixes need to be back-ported to these features, the same rules described in stable_kernel_rules.txt are applied. (Please refer Appendix A2 for stable_kernel_rules.txt.)

The basic ideas used to define the rules are 1) fairness to the industry, 2) stability and quality of the LTSI tree, and 3) value to the industry. Since the LTSI tree should be ready-to-use by the industry, it needs to be highly stable. Although the LTSI chief maintainer has a responsibility to decide which features will be back-ported, he/she needs to consult with the CEWG Architecture Group to prioritize such features, and with an LTSI supervisor to understand the complexity and assess the impact on the stability.

Detail rules will be defined by the LTSI project office, and the LTSI chief maintainer. Interviews with BSP authors will also be conducted.

For back-porting of bug fixes, all fixes applied to the community's long-term-stable tree will be incorporated into the LTSI tree.

5.2 Feature integration rules (LTSI Staging)

The same rules for the linux-staging tree are applied here. (Please refer Appendix A3 for linux-staging.) In addition to those, following rules are also applied.

- the code needs to be reviewed by upstream maintainers before submitting to the LTSI staging tree
- there is a reasonable reason why the code is not currently incorporated into upstream
- the author still has an intention to upstream the code

Although non-upstreamed code is accepted with the above conditions, this is not a place to dump out non-upstreamed code. The feature needs to be very useful for the industry, and is still expected to be upstreamed in the future.

Selected features in the LTSI staging tree will be pulled to the LTSI tree at the initial creation of the LTSI tree from the community's long-term-stable tree. Although the LTSI chief maintainer has a responsibility to decide features to be pulled, he/she needs to consult with the CEWG Architecture Group to prioritize the features, and with an LTSI supervisor to justify the selection.

5.3 Fix/Feature acceptance rules (from industry engineers)

As described in section 2.4, the LTSI project accepts fixes and features for the LTSI tree from industry engineers. Then, an LTSI upstreaming engineer works with an appropriate upstream engineer to upstream the fixes or features, as well as to incorporate these into the LTSI tree if the fixes or features are worth being incorporated. This is to accelerate contributions to upstream from the industry. Patches sent to be incorporated into the LTSI tree must follow the Documentation/SubmittingPatches rules. In addition to that, the author needs to agree to respond to inquiries from an LTSI upstreaming engineer if re-implementation to the latest kernel is required.

6 LTSI Value Proposition

The value propositions of the LTSI for ecosystem partners are summarized as follows.

For the Consumer Electronics Industry

- Efficient and less-expensive operations in enabling Linux to a variety of devices with less fragmented use of Linux
  - Easy to enable Linux on a variety of devices with less resources
  - Easy to apply latest bug fixes, and determine problems.
    - As the differences between the LTSI tree and the latest version of upstream tree are expected to be small due to feature back-porting, industry engineers can work with community engineers to test and fix on the latest version easily.
- No need to apply some local fixes anymore
  - Some local fixes will be either incorporated into upstream through the industry staging tree and back-ported into the LTSI tree by the LTSI BP team, or, directly incorporated into the LTSI tree and upstreamed by an LTSI upstreaming engineer.
- Additional features which are beneficial for the industry are available in the LTSI tree
  - The LTSI project accepts features which are not incorporated into upstream yet but are very beneficial for the industry, and continuously works with an author to upstream it. Therefore, the industry can use optimized Linux for the industry.
- More close interactions with upstream engineers
  - Through this project, the LTSI team encourages industry engineers to work with upstream engineers, and connects both.

For the Semiconductor Industry

- Enhancements and bug fixes in upstream will be propagated to the LTSI tree, then, applied to distributions and actual embedded systems. This is a very effective way to enhance and maintain their code.
- Semiconductor vendors can directly contribute to the LTSI tree to integrate latest features, and creates an LTSI based BSP which is ready to be merged into the current product software stack by the set vendors.
- No need to work on an individual tree inside customers privately to merge with the BSP for newly designed SoC.

For Distributors

- As the LTSI Back Port Team maintains the LTSI tree up-to-date, distributors can focus on their differentiations.

For the Linux Community

- Better communications with industry engineers with a common code base.
- More contributions are expected from the consumer electronics industry.

7 Summary

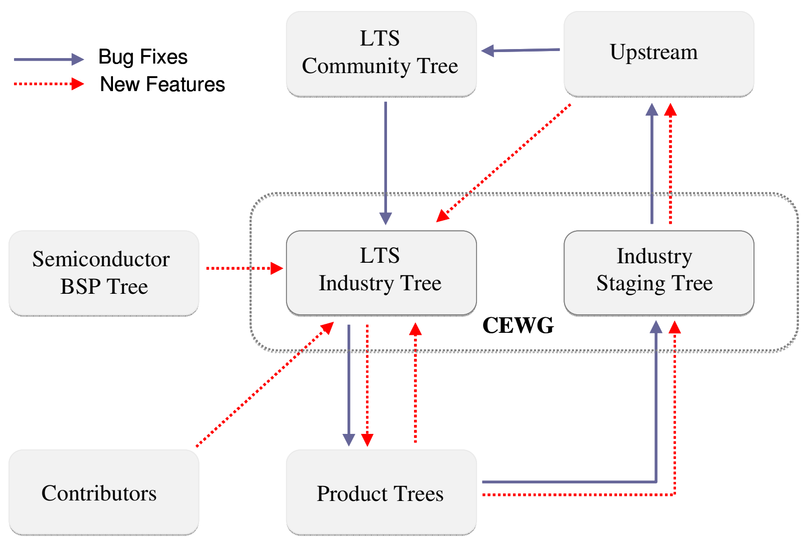
The LTSI project is the first instance to bridge between the community and the industry. It focuses on the consumer electronics industry, at first, but can be extended to other industry by using the same infrastructure. By creating the LTSI tree, a new and more effective ecosystem for the industry is expected to be formed.

The LTSI project team continues to work with the community and ecosystem partners to make this project be more efficient, effective, and influential.

Appendices

A Supplemental Material

A.1 Mature Model for Operation Flows



Eventually, as the industry matures in working with the community, the Upstreaming Team and LTSI staging team will be no longer necessary. The operation flows are simplified as shown in Figure A-1.

If industry engineers are matured further, then, the Industry Staging Tree and associated resources such as an integration consultant will no longer be necessary.

On the other hand, there will still be some features which can not be merged into the Upstream due to the uniqueness of the embedded system development. CEWG still needs to accept features from several groups, incorporates them into LTSI tree for the industry.

A.2 stable_kernel_rules.txt

```
Everything you ever wanted to know about Linux 2.6 -stable releases.

Rules on what kind of patches are accepted, and which ones are not, into the
"-stable" tree:

 - It must be obviously correct and tested.
 - It cannot be bigger than 100 lines, with context.
 - It must fix only one thing.
 - It must fix a real bug that bothers people (not a, "This could be a
   problem..." type thing).
 - It must fix a problem that causes a build error (but not for things
   marked CONFIG_BROKEN), an oops, a hang, data corruption, a real
   security issue, or some "oh, that's not good" issue.  In short, something
   critical.
 - New device IDs and quirks are also accepted.
 - No "theoretical race condition" issues, unless an explanation of how the
   race can be exploited is also provided.
 - It cannot contain any "trivial" fixes in it (spelling changes,
   whitespace cleanups, etc).
 - It must follow the Documentation/SubmittingPatches rules.
 - It or an equivalent fix must already exist in Linus' tree (upstream).

```

```

Procedure for submitting patches to the -stable tree:

 - Send the patch, after verifying that it follows the above rules, to
   stable@kernel.org.  You must note the upstream commit ID in the changelog
   of your submission.
 - To have the patch automatically included in the stable tree, add the tag
     Cc: stable@kernel.org
   in the sign-off area. Once the patch is merged it will be applied to
   the stable tree without anything else needing to be done by the author
   or subsystem maintainer.
 - If the patch requires other patches as prerequisites which can be
   cherry-picked than this can be specified in the following format in
   the sign-off area:

     Cc: <stable@kernel.org> # .32.x: a1f84a3: sched: Check for idle
     Cc: <stable@kernel.org> # .32.x: 1b9508f: sched: Rate-limit newidle
     Cc: <stable@kernel.org> # .32.x: fd21073: sched: Fix affinity logic
     Cc: <stable@kernel.org> # .32.x
    Signed-off-by: Ingo Molnar <mingo@elte.hu>

   The tag sequence has the meaning of:
     git cherry-pick a1f84a3
     git cherry-pick 1b9508f
     git cherry-pick fd21073
     git cherry-pick <this commit>

 - The sender will receive an ACK when the patch has been accepted into the
   queue, or a NAK if the patch is rejected.  This response might take a few
   days, according to the developer's schedules.
 - If accepted, the patch will be added to the -stable queue, for review by
   other developers and by the relevant subsystem maintainer.
 - Security patches should not be sent to this alias, but instead to the
   documented security@kernel.org address.


Review cycle:

 - When the -stable maintainers decide for a review cycle, the patches will be
   sent to the review committee, and the maintainer of the affected area of
   the patch (unless the submitter is the maintainer of the area) and CC: to
   the linux-kernel mailing list.
 - The review committee has 48 hours in which to ACK or NAK the patch.
 - If the patch is rejected by a member of the committee, or linux-kernel
   members object to the patch, bringing up issues that the maintainers and
   members did not realize, the patch will be dropped from the queue.
 - At the end of the review cycle, the ACKed patches will be added to the
   latest -stable release, and a new -stable release will happen.
 - Security patches will be accepted into the -stable tree directly from the
   security kernel team, and not go through the normal review cycle.
   Contact the kernel security team for more details on this procedure.

```

```

    Review committee:

     - This is made up of a number of kernel developers who have volunteered for
       this task, and a few that haven't.
```

A.3 linux-staging

```
PURPOSE

The linux-staging tree was created to hold drivers and filesystems and
other semi-major additions to the Linux kernel that are not ready to be
merged at this point in time.  It is here for companies and authors to
get a wider range of testing, and to allow for other members of the
community to help with the development of these features for the
eventual inclusion into the main kernel tree.

This tree will be included in the daily linux-next builds, and will get
testing by all users of that tree.

The rules of what can be included here is as follows:
        - the code must be released under a Linux kernel-compatible
          license
        - the goal of the developers must be to merge this code into the
          main kernel tree in the near future, but not for the next
          kernel release.
        - the code must build properly on the x86 platform
        - this is not a tree for bugfixes or rewrites of existing kernel
          code, this should be for new features, drivers, and
          filesystems.
        - the patches included must detail exactly what is needed to be
          completed in order for them to be included into the main
          kernel tree.
        - there must be some email address associated with the patch
          that can be used for bug reporting and questions about
          cleanups and testing the code.

What this tree is not:
        - it is not a place to dump features that are being actively
          developed by a community of people (reiserfs4 for example.)
        - it is not a place to dump code and then run away, hoping that
          someone else will do the cleanup work for you.  While there
          are developers available to do this kind of work, you need to
          get someone to agree to "babysit" the code.
```

B List of Acronyms and Glossary

BSP	: Board Support Package
CEWG	: Consumer Electronics Workgroup in the Linux Foundation
IAB	: Industry Advisory Board
LTS	: Long Term Support
LTSI	: Long Term Support Initiative
LTSI Tree	: Long Term Support Industry Tree
MM	: Memory Management
PM	: Power Management
RT	: Real-Time
SME	: Subject Matter Expert
SoC	: System on a Chip

End of Document