**Part 1 (Kick Off & Overview)**

July 10, 2023, 3:35PM

58m 1s

 **Mohit Gupta** 0:03  
OK, so to start I just want to uh, lay down the requirement and the goals that we have.

 **Mohit Gupta** started transcription

 **Mohit Gupta** 0:13  
Uh, so since we have been having a few incidents lately in me pub and we want to get better at that.  
So that's why we are doing this review.  
Some of the goals of this review series is to improve the reliability and stability of media publishing pipeline in general, and of course, when we improve the reliability and stability, we will reduce the number of incidents happening.  
The primary goal, I would say, includes improving the DRI experience that we are giving to the Swiss and improve their work load balances.  
We we've seen people working over weekends, late nights, working long hours in escrow week, so that's that's all that we want to uh, I get better at then during this review, wherever we find opportunities, we will, we will take them and we will, we will redesign, refactor or rewrite whatever is required for the longer term, not not as a quick fix night and then the goal is not to find the person responsible for any issue that is found.

 **Yashasvini Rathore** joined the meeting

 **Mohit Gupta** 1:28  
It could be a years old issue which we are finding out now, or it could be an issue which was caused by by a hot fix which was done in an escrow.  
Week and we all understand the pressure of timelines and if Media Publishing does not happen, it just breaks the whole flow of keeping customers protected and productive, right.  
So we understand.  
No, the the time pressure, there are more challenges in media publishing.  
I'll cover in the next slide one of the.  
Biggest one, as I already talked about is the load that Media publishing team gets in the escrow week.  
It almost all the load comes in the escrow week sometimes.  
Uh in the weekend, just before the the past Tuesday.  
Right.  
So that is one of one of our big challenges then the scale, the the size of files that we deal with and the number of such files, it's just huge, right?  
Isos are 5 gigabytes each.  
VSTS are 120 gigabytes, right?  
That is huge.  
So that scale is another challenge, and most of our incidents since they occur in the escrow week.  
So we have to treat them as SEV tools and work accordingly.  
And then some of the hotpatching has to be done during the escrow week.  
I'll pause here for any questions, comments or anything that you want to add.  
Anyone wants to add on our goals for this exercise and any challenges that you want to add for media publishing, anyone?

 **Achal Shah** 3:24  
I think also also the all the different types of endpoints that people was to publish to, they're all very different, right?

 **Mohit Gupta** 3:24  
Yes, I do.  
Yes.  
Yep.

 **Vaithi Krishnasamy** 3:41  
One other thing I I'm able to remember is Media doesn't have the concept of TTGL before.  
I'm not sure now.  
Uh, so it was like, difficult to stage things, you know, at the end point before and then wait for TTGL for them to make it live was not possible.  
I'm not sure if that's still the case.

 **Mohit Gupta** 4:02  
Yes, that is still the case.  
I think I've covered in the next slides, but yeah, that's another good point.  
We we can only start after SMF signs off.

 **Vaithi Krishnasamy** 4:07  
Yes.  
Yep.  
And the this has been like I think it was discussed before, you know whether we wanna introduce TTGL, but I don't think the conversation never reached any fruitful stages here because partners has to support you know the various endpoints which Archill was saying right then everyone has to support the concept of TTGL.

 **Mohit Gupta** 4:37  
Sure we can.  
We can talk about it in our solutioning phase.

 **Vaithi Krishnasamy** 4:43  
Thanks, mom.

 **Mohit Gupta** 4:49  
Anything else?  
Good.  
So just like any other initiative, we also have a V team and if you see your name on the screen right now, you're part of the V team.  
This is your time to exclude yourself.  
Excuse yourself from now on, this is the V team and we are all responsible for making this a success.  
I'll wait for like 15 seconds.  
No.  
So good.  
Thank you, everyone who's on the screen.  
Thank you for volunteering for being the being part of the V team.  
So now let's talk about some of the things that we want in the system.  
Some some of the design principles.  
Uh.  
In Media Publishing, we've already talked about the scalability, the size that we, that we deal with due to any fixes coming out of this exercise.  
We cannot just cannot compromise on scalability.  
If we can, we'll target to improve it, but scalability is one of our top most priorities, right?  
And then we do a lot of things parallelly wherever possible because of code to customer or whatever reasons.  
If we if we can identify any.  
Improvements in efficiency in the time it takes to publish, we will introduce more parallelism.  
Uh, unless it is causing an issue, we will not.  
Ohh we will not reduce the existing parallelism in the pipeline.  
Then much of the communication with the outside world and within the internal subsystems of media publishing, it's always synchronous.  
We will identify.  
Opportunities to make things even more asynchronous to make things more fault tolerant if another system is down, it should be able to resume when it comes back up.  
For example, yes Niraj.

 **Dhiraj Singh** 7:10  
It might actually.  
This looks like say we are trying to design a new system, right?  
We are trying to set up the principles and then thinking the issues will encounter.

 **Mohit Gupta** 7:17  
He.

 **Dhiraj Singh** 7:21  
This is how we'll redesign, I will say because the goal is really understanding and simplifying DRI, work life and all those things, right?  
So I think we should look at the incident 1st and then because media was designed with the design set of design principles, some of them might not be valuable anymore because those decisions are made four years back, right?  
So I think we we can look at the top incidents, we can try to frame it back.

 **Mohit Gupta** 7:45  
In.

 **Dhiraj Singh** 7:50  
What are the core issues and why are they existing design principles are not working and whether we need to tweak it or not because this I think it's starting from this position.  
Seems like we are saying hey, let's go and redesign media.

 **Mohit Gupta** 8:06  
Umm, you're right.  
Uh, I agree.  
We are objective is to.  
Address incidents and we will do that.  
The reason I brought it up was to just bring everyone on the same page that this is everyone's understanding of the existing system.

 **Dhiraj Singh** 8:27  
So these are design principles of existence.  
Where are design principles of existing system?  
Let's start from there.  
If we have to do that.

 **Mohit Gupta** 8:33  
So my understanding is this is these are all the principles that existing system follows.  
We will better it and due to any fixes we will not go back on any of these principles.  
This is what my understanding is.

 **Dhiraj Singh** 8:46  
Bug.

 **Mohit Gupta** 8:48  
Anyone disagrees, please feel free to add.

 **Dhiraj Singh** 8:56  
Media experts, do you think these are the high level design principles?  
We have already tried to implement.

 **Mohit Gupta** 9:10  
Silence says yes, maybe.

 **Dhiraj Singh** 9:15  
I will think so little next Gen was designed like in in this model we wanted to be scalable parallel.

 **Mohit Gupta** 9:19  
Yep.

 **Dhiraj Singh** 9:21  
I think all of these right different implementation.

 **Mohit Gupta** 9:22  
And when we do the deep dives and when we do the deep dives, we'll, we'll we'll validate these.

 **Dhiraj Singh** 9:28  
Sure.

 **Mohit Gupta** 9:30  
Yeah.  
Hopefully that answers your question Viraj.

 **Alexey Loginov** joined the meeting

 **Dhiraj Singh** 9:34  
Umm.

 **Achal Shah** 9:36  
Hey, wait one quick question before you go.

 **Mohit Gupta** 9:36  
Some extent.

 **Achal Shah** 9:41  
So, you know, yet the the V team.

 **Mohit Gupta** 9:41  
Yeah.

 **Achal Shah** 9:43  
Right.  
And then there is you have the the media dev teams in Redmond and IDC, right?

 **Mohit Gupta** 9:45  
Umm.

 **Achal Shah** 9:51  
So I just maybe you'll address this later, but just curious about how we worked because if we thought about that or interactions?

 **Mohit Gupta** 10:02  
That's a that's a great point.  
Actually I missed it during the meeting.

 **Achal Shah** 10:05  
Yeah.

 **Mohit Gupta** 10:06  
It's like yes, so.  
When I say we team, these are the people who would, uh be kind of the experts and who will validate these design principles and maybe sometimes validate the way current media publishing system works.  
For example, if we have.  
To ask any question about how media orchestration service works, we will go to tavishi night.  
If we have to go to uh, anything about partner central or publishing in general, I think polo is the best person to contact, right?  
So that's how I included the SMEs in the in the suite team and also the expertise and the experience of how how good systems are designed and work in general and in UPS in particular.  
So that's the idea of this meeting.  
Then outside of this meeting, we have the media publishing team.  
People from that team, including IDC, will be presenting how the current system works.  
What is their understanding of the current incidents?  
The current DRI experience and what they think is an issue in the system right now when they bring a class of incidents and a a potential issue in the system, the V team will come together and hash out everything, whether whether the current system follows all the design patterns.  
Is there any opportunity to to better it and if yes then should we redesign, refactor, rewrite, whatever so this is kind of the V team that in my mind takes the decision what to do collectively and Media publishing team outside of this V team will help the V team arrive at no.  
At least the issues in the current system.

 **Achal Shah** 12:14  
Cool. Thanks.

 **Mohit Gupta** 12:15  
And and if anybody present in this call or listening to the recording later in the media publishing team wants to join this V team, you're most welcome.  
Uh, but Yep.  
Hopefully that answers your question, Achal.

 **Achal Shah** 12:38  
Yep, thanks. Yeah.

 **Mohit Gupta** 12:41  
Yep, so I I won't bore you with each and every point.  
I think we all agree on these design principles. Uh.  
Now going forward with the next slide, so we just want to cover media publishing at a very high level today.  
Umm, so in in maybe 30 seconds if I have to describe media Publishing Media Publishing creates an intent which is sent to the SMF system which then creates the media, validates it, signs off and when it signs off.  
That's when it comes back to media publishing for ingestion, and some of these this happens through the I publish API, right?  
And when it comes back to media publishing system, that's when.  
We ingest, scan, approve and go live right now to people who are not in media publishing.  
We thought it was a good idea to give them a very high level of what happens behind the scenes when SMF sends us the sign off.  
What happens behind the scenes so?

 **Dhiraj Singh** 13:51  
Yeah.  
What?  
What does sign off mean and what are the activities there?  
And I want you to understand if there are opportunities to trigger something even before sign off.

 **Mohit Gupta** 14:03  
Yes.  
OK, so maybe let me move forward.  
One slide and they'll come back to it.  
So there's something called an intent and and intent is just a Jason file which contains 3 things.  
It is generated by the media orchestration service in AWE.

 **Shailja Joshi** joined the meeting

 **Mohit Gupta** 14:22  
This intent is sent to the SMF system, yes, Achal.

 **Achal Shah** 14:29  
No.  
What?  
While I was going to say just in, I think it's helpful to if you think of the intent, it's very similar to what AWE does for unified packages, right?  
So unified packages have a bunch of child packages, right?  
Which go into it.  
Umm.  
Or maybe it's the I'm thinking of the the bus bus model, right?  
Same way for media you have a bunch of packages that would go into creating the media.  
So every time one of these packages churns, that's when the intent is created.

 **Dhiraj Singh** 15:09  
Yeah, but I thought there was some manual activity which is done which could take some time and then only some sign off happens.  
So are there?

 **Achal Shah** 15:18  
No, no, no.  
So that it so sign off is afterwards, right sign off is after the media is created.

 **Dhiraj Singh** 15:21  
Umm.

 **Achal Shah** 15:23  
This is just kicking off media creation.  
The intent.

 **Dhiraj Singh** 15:27  
I see.  
I see intent OK.

 **Achal Shah** 15:27  
Yeah, yeah, yeah.

 **Dhiraj Singh** 15:28  
Intent like ohh like UPS calling into external partners to trigger the media UPS.

 **Achal Shah** 15:34  
Correct.  
Yeah.  
Yeah, very similar to how we do for.  
Problem I guess it is unified right?  
And that's what I'm thinking of, yeah.

 **Dhiraj Singh** 15:43  
I see. OK.

 **Vaithi Krishnasamy** 15:45  
So, Achal, so how I'm maybe again out out of date and not up to date, but when this media publishing started, right, so how the SMS team used to operate is they used to build a media you know before even like you know sending out an intent or the metadata to publish our for us to ingest, right.  
So at that point of time, they used to do some testing, some basic testing before they even, you know, send out this intent or or for us to even consume the media. Right.  
So I think they do some boot tests and all those things before you know, even the publishing request reaches our system.  
And the problem at that time was, you know, because the even the media was generated only at the end moment, usually on the weekends and then they do these basic testing before triggering the publishing.  
So the time you know the publishing request used to reach the media side, I mean like the this Me pub side used to be like very late in the game.  
I want to know like if that is still the case or or you know that site has been improved.

 **Achal Shah** 16:49  
No, I think the intent like media creation is happens through the cycle right through our as churns come, it happens right.

 **Vaithi Krishnasamy** 16:55  
Yeah.  
Mm-hmm.

 **Achal Shah** 16:59  
But anyway, I think Mohit will go through.  
So there's the intent part which is before and then there is the publishing part which comes.

 **Vaithi Krishnasamy** 17:04  
Yeah.  
Yeah, I'm.  
I'm talking more on the publishing part because that's when you know most of the issues show up, yeah.

 **Achal Shah** 17:08  
Yeah. Yeah, yeah.  
Yeah.  
So Mohit, I think we'll talk on that, yeah.

 **Aravind Siddoju** 17:18  
Mohit Year and mute if you're.

 **Mohit Gupta** 17:21  
Study Sunny so most of the communication happens on a service bus topic.  
We send the intent to SMF on a service bus topic.  
They create media, they perform the validations, then they sign off and then sends the sign off over another service bus topic and then media publishing setup and starts with the ingestion and publishing.  
That's the overall flow.  
Any yes, Antonio.

 **Antonio McMichael** 17:51  
Yeah.  
When it comes to sign off, what is the SMF sign off?  
Is that something they do where they validate the integrity like a BBT or is it something else?

 **Mohit Gupta** 18:01  
Yes, yes, it's something like.  
So they have a bunch of tests and they have, uh, some requests.  
Especially so.  
For example, there's one issue today where they validate their boot up a VM using that media which is which is generated and they test the basic tests and any new feature requests.  
Once that is done, some of them is manual.  
Once that is done, they send the sign off.  
So yes, SMF teams team performs that validation after creation of media.

 **Antonio McMichael** 18:39  
Got it. Thanks.

 **Dhiraj Singh** 18:41  
And when we say media, the composition is a bunch of files, right?  
Like, how does the composition of files look like for a media player?

 **Mohit Gupta** 18:48  
It depends on the file type, so if it's a of for example VSD, there are different types of files that we received.  
There is VHD.  
There is.  
DVST then CVST.  
Those are all part of the same media, but they're different.  
But you can say different media which does the same thing.  
So VSD is the full VSD, DVD is the Delta VSD.  
So if they have a base VST, they can create the new media using that delta VH D which is much smaller.  
ISO is, I think, a single file.  
Umm many of them are not very not a lot of files, a few files, a meat experts please.

 **Dhiraj Singh** 19:34  
I see. OK.

 **Mohit Gupta** 19:37  
Anyone wants to add anything here?

 **Deepak Kunwar** 19:42  
So wait one thing, So what data just said so directs what happens to actually we also create media based on the language and architecture.  
For example, for only one payload and if there are 10 languages and 2-3 architectures, so it will be 10 into 330.  
Media needs to be created so language plus architecture also plays an important role to decide that how many media needs to be created.  
So these are two important parameters for media creation.

 **Dhiraj Singh** 20:10  
So when you say V means SMF, team creates it, right?

 **Deepak Kunwar** 20:14  
Yeah, SMF team creates.

 **Aravind Siddoju** 20:14  
Like.

 **Deepak Kunwar** 20:15  
Yeah, but the the source of truth or the number of images is passed from the one pub team we provide them that you need to create her team images or 50 images and what will be the architecture but with the language so that is provided by one per.

 **Dhiraj Singh** 20:23  
OK.  
So if it is coming later then it's fine, but I wanted to understand say I'm assuming media creation itself takes some time.  
Maybe I don't know how half a day, right?  
And then there must be stages that are based base image is created.  
Then you start creating language and right and then I'm assuming that the final validation.  
I'm sure we don't validate everything.  
You only validate one of them, right?  
So I I'm curious to understand that whole process as well as part of this exercise, we're gonna think that can help drive some of our improvements.  
Uh, before we even do any activity on our side.

 **Deepak Kunwar** 21:13  
That.

 **Mohit Gupta** 21:14  
Sure.  
So we'll we'll maybe have somebody from SMF give us a 1520 minute walks of that.

 **Dhiraj Singh** 21:25  
Hmm.

 **Mohit Gupta** 21:25  
It's uh, so moving on, I think let's talk about image list from this slide.  
Thanks Deepak, for reminding me.  
So image list is a list of media that needs to be created and this comes from something called source of truth which we store in one pub so.  
Although Media team creates the media, but we tell them why I an intent.  
What you need to create and what payload to use to create that media.  
So for example, we create an LSU and LSU this version using this job ID.  
You have to create the media that also we give.  
So three uh, an intent has three basic things.  
The basic release details which release what update type, which product then under that release, that update type that product what all image lists we need to create and using what payload.  
That's the purpose of this intent.  
Yep.  
I'll go back one slide and.  
Maybe I'll let Arvind talk about what happens behind the scenes in I publish across.  
I publish, I mean behind I publish.

 **Aravind Siddoju** 22:52  
Umm, shut up.  
Let me share my screen actually.

 **Mohit Gupta** 22:58  
Yeah, let's stop sharing.

 **Aravind Siddoju** 22:59  
So yeah, part of this first session.  
Hi everyone as part of this session first session I just want to give some brief overview on like what happens on the I after I publish like since we know every reaches high publish endpoint even to reach all the one of the systems.  
Now I just want to particularly focus on what happens on the media side and how of the request from AWE eventually reaches the the core business logic of the media services.  
And then I'll walk you through the the solution, the services that we have and how they differ and from what I'm showing in the high sorry high level diagram over here and then maybe we can talk more on something comes up in our discussion.  
So I'll I'll just cover this one higher level diagram before I jump in further.  
So this is in a nutshell how the three systems interact.  
I hope many of you might have already seen this, but uh. Uh.

 **Dhiraj Singh** 23:57  
Yeah.  
Arvind, can you please zoom?

 **Aravind Siddoju** 23:57  
I'm gonna.

 **Dhiraj Singh** 23:58  
Zoom one level that will help me.

 **Aravind Siddoju** 24:00  
Asher. Actually.

 **Dhiraj Singh** 24:01  
Thank you.

 **Aravind Siddoju** 24:02  
Yeah. Let me know if you can see it.

 **Dhiraj Singh** 24:04  
Yeah, this is fine.

 **Aravind Siddoju** 24:05  
I don't have the exact be a picture with me telling me OneNote.  
Sorry for that.

 **Mohit Gupta** 24:11  
Maybe can you zoom your screen if you control scrolling?

 **Aravind Siddoju** 24:15  
Yeah, that's fine.  
I think I think I can can zoom the image.  
Let me know if you can see it.

 **Dhiraj Singh** 24:19  
Yeah, this is good for me, yeah.

 **Mohit Gupta** 24:20  
Is better.

 **Aravind Siddoju** 24:21  
Is good, right?  
Right.  
So so everything starts from AWE here, which basically orchestrates everything.  
We do have a place, place contains stages.  
I'm not going to speak in detail on the AWE side of things, but I'm gonna give some reduction on how things work.  
So basically how this play for the each update type for product and it has its own set of stages and its own set of activities within the stages which are expected to kick off depending upon some signals on a higher level.  
We basically AWE basically starts everything when the BBT and the CAC completes for the payload, the actual LCU payload, the packages that we generate.  
So once that happens, basically AWE asks the one pub system.  
So although it's one pub, so in this current discussion, it's entirely me.  
Pub.  
I'm going to refer so terribly.  
Basically, cars and API of Media publishing team to provide the list of images that it want to be created from the SMF team.  
So it just provides that update type details, release information and a few other details.  
And we on the ME pub side store that information per product per update type the the number of or the list of images that needs to be created and architectures that are supported for those images.  
So we call them drops on our site.  
Each drop is and each image and we do have bunch of architecture.  
So eventually SMF team will create number of images that are mapped with this drops and architectures and languages.  
So we just provide those details as soon as.  
AWE oh oh sends a request asking us the details of the so source of truth, which is basically the list of images that needs to be created for that update type.  
So once AWE receives that so it sends a service bus message.  
At this stage, we call that the service bus message as intent, so that would be an input request to the SMS system.  
So intent basically contains the payload for which we need to create the media and the the list of media.  
The list of images that we need to create for that update type so that request will be sent to SMS system.

 **Shailja Joshi** left the meeting

 **Aravind Siddoju** 26:54  
So there are a bunch of signals that gets handshakes between the AWE and SMF.  
So as soon as so everything is like service bus message system between AWE and SMF.  
So at the moment it's send it, it would be request sent notification and then once they acknowledge it, it would be some request acknowledge and once they finish the media umm completion of the media creation and once they send a signal back telling the media.  
Completion creation is done so that that we call as intent sign off the moment the SMF team signs off the intent.  
So that's when AWE kicks off the publishing process for the media.  
That's when AWE kind of calls.  
It's individual activities which basically calls different endpoints or different workflows within one pub system.

 **Vaithi Krishnasamy** joined the meeting

 **Aravind Siddoju** 27:45  
Although it's a single line in the next the diagram I'll show like how that differ.

 **Dhiraj Singh** 27:48  
Yeah.  
So are there multiple in intent sign off or one sign off for all channels?

 **Aravind Siddoju** 27:55  
Uh, so one intent per product per update type?

 **Dhiraj Singh** 28:00  
Product, they're multiple sign offs is comedian I see.

 **Aravind Siddoju** 28:02  
Right.

 **Achal Shah** 28:04  
I believe sign off is on the intent, right?

 **Aravind Siddoju** 28:08  
Like.

 **Dhiraj Singh** 28:11  
So once I sign off has come suppose the intent was for VHD.  
So once the sign off has come, there are no more activities on these blue boxes like it's all done from SMF.

 **Aravind Siddoju** 28:16  
Umm.  
Right.  
These three blue boxes will be done.  
So SMF basically sends a stores the information off that media in this blue boxes and they basically share those drop locations.  
Uh to us in the in the response back we shared apply kind of returns like calls with those details to the one pub system.

 **Dhiraj Singh** 28:45  
OK.

 **Aravind Siddoju** 28:49  
Mohit, you have your hand raised.

 **Mohit Gupta** 28:51  
Tony, you just say it so uh blue boxes remain.  
Ohh.  
I mean they they they stopped functioning.  
But there's just one API call that we make for mission about the median.

 **Aravind Siddoju** 29:01  
Right.

 **Fatuma Omer** joined the meeting

 **Aravind Siddoju** 29:06  
Umm, Alexey, you have your question?

 **Mohit Gupta** 29:07  
Media seeker media seeker API.

 **Aravind Siddoju** 29:10  
I'm just gonna cover that in a minute.  
Uh, let's see how your hand is.

 **Alexey Loginov** 29:17  
Yeah.  
Morning, you guys.  
Just quick question.  
I'm confused about direction of some of the blue errors on this diagram.  
In particular, can you please explain the error that goes to one pub from from media related stuff?

 **Aravind Siddoju** 29:28  
OK.

 **Alexey Loginov** 29:36  
Yeah, this one.  
The big one?

 **Aravind Siddoju** 29:37  
This blue boxes, right?  
So OK.

 **Alexey Loginov** 29:39  
Yeah, so this error that goes to one pub, what does it represent?

 **Aravind Siddoju** 29:42  
Right, right.  
So they didn't.  
They don't directly go this way, so basically like let me take a step back.  
So uh, one SMS creates the media, it just stores in these different channels and forget about this blue line for now.  
Once SMF comes back telling everything is done, AWE sends the details back to one pub system and we do have the ohh a workflow called induction which kicks off immediately after intense get signed off.  
That's when one pub actually tries to retrieve the media information from these storage other way around.

 **Alexey Loginov** 30:19  
OK.  
So just has to be other way around.  
One pub knows and retrieves information from different media storages, right?

 **Aravind Siddoju** 30:22  
Right, right, right.  
Right.  
But thank yeah, I'll thanks.

 **Alexey Loginov** 30:29  
So has to be diverse.  
Mm-hmm.  
Thank you.

 **Aravind Siddoju** 30:32  
Thanks for that.  
Uh, any other question?

 **Achal Shah** 30:37  
Yeah, I think it's also worth going on.  
There's a question Antonio had asked in chat.

 **Aravind Siddoju** 30:41  
Uh-huh.

 **Achal Shah** 30:42  
Why is 1 pub the source of truth?  
May maybe worth going over that here.

 **Aravind Siddoju** 30:48  
Yeah, so this this has been the long discussion of question that we've been trying to solve.  
So right now, umm SMS had its own source of truth, but although they they rely on what we send in the request, AWE kind of maintains there is one little layer of source of truth and that side related to media then we do have the actual source of truth which we use to create the media.  
Uh, which is basically sent in the actual request to create a media.  
So that that's been a lot of confusion for some time now on uh, who need to own the source of fruit, at least on the one pub side.  
I feel it's it's one pub team who should own that because ultimately we are the one who's publishing it and and we know what the publish.  
But yeah, again there there's been some discussion we had.  
I mean, it's been some internal discussion.  
We've been discussing it tavishi and few folks from AWE and like a white could, it could not be any other way around where one pub should not own anything and AWE can own the source of truth.  
Umm.  
So yeah, there are there are multiple things that's going on and that's that's again what, uh topic to bring up for discussion maybe in the upcoming sessions, but yeah, but right now, yeah, this is the current situation.  
One pub is currently the source of truth.  
Uh, which provides the list of images that need to be created.

 **Mohit Gupta** 32:26  
But the but this is a great question.  
Thanks Antonio for for asking that.  
But I would take it a step further. Maybe.  
Why even UPS or why even SMF?  
Shouldn't we have a a a common service WSD wide which knows hey for this release for this update type, what's the media?  
That's that means to be generated and and it need not to be owned by one particular team because it's not, it's not one particular team relevant, it is relevant to WSD in general, on maybe even higher.

 **Dhiraj Singh** 33:04  
I'm assuming this is not like the source of incident because something similar we do in UPS as well, because the publisher is closer to the partner team, the requirements come to the part like us.

 **Mohit Gupta** 33:11  
Yep.

 **Aravind Siddoju** 33:15  
Like right?

 **Dhiraj Singh** 33:16  
So even for you, we actually hold the chunks.

 **Mohit Gupta** 33:17  
Umm.

 **Dhiraj Singh** 33:20  
In that case, we don't need to send back the chunks to AWE because AWE doesn't need to know.  
But I think in this case you AWE needs to know to what to trigger and SMF needs to know.  
So I think in the with that perspective seems like it is set up fine for now and I don't feel like is it do we think it's causing incidents or or not?

 **Aravind Siddoju** 33:42  
Well, uh, I'll I'll go other way around.  
Directs.  
But let's say if we want to onboard a new update type for a product, or maybe when we are onboarding a new product for an update type.

 **Dhiraj Singh** 33:49  
Umm.

 **Aravind Siddoju** 33:55  
So there's always been condition confusion and we we have to maintain this consistency and like whoever is holding these.  
Individual source of truth and systems have their validations right?

 **Dhiraj Singh** 34:04  
Yeah.  
Agree.  
Agree and onboarding.  
So I'll tell you like we tried to actually create one common service, the fundamental team try to create it didn't go anywhere, right?

 **Aravind Siddoju** 34:13  
I see. OK.

 **Dhiraj Singh** 34:16  
So it's a hard problem to solve reading a common service which everyone understands, right?

 **Aravind Siddoju** 34:19  
Right.

 **Dhiraj Singh** 34:22  
So that's where we are stuck.  
Also in UUP land where we just own and there's a manual transaction.  
So I agree that problems will come at onboarding time for incidents.  
But once you are onboarded, I think then from there on you should be like consistent.

 **Aravind Siddoju** 34:36  
Like.  
Yeah.  
Yeah, I think ohh.  
Source of truth necessarily doesn't cost too many incidents like once it's stabilized.  
But yeah, again, that's a matter of concern.  
A paper multiple teams winning source of truth.  
But yeah, we can.  
We can bring that in upcoming discussions.  
Umm any other questions before I move on so it some more details.  
I like so again, so on the media side, we do have a multiple workflows.  
I'll just start that in the next image, but this system basically has tie up with all the endpoints which we published to for different update types.  
So we we publish to MCR ACR for container and then publish to marketplace and Stack gallery for VHD and we published 2 apps.  
Uh, so reduct share for APPF and we similarly publish to VLSC using for the ISO images using the ESRP portal.  
So there are different umm endpoints different endpoints which are not unique which are even the design and implementation where they are completely different that that's uh those that are actually the different systems and services that we talked to from the media.  
Of the system itself, I just jump to this.  
Another kind of a a diagram kind of picture.

 **Dhiraj Singh** 36:12  
Yeah.  
What question of the diagram about like, umm like we have the payload in these blue boxes?

 **Aravind Siddoju** 36:16  
Uh-huh.

 **Dhiraj Singh** 36:19  
Does one pub process it further or like what kind of metadata it adds?

 **Aravind Siddoju** 36:20  
OK.  
Like.

 **Dhiraj Singh** 36:24  
Or is it significant or it's more of a straight pass through, just like clubbing them together and going to the endpoints?

 **Aravind Siddoju** 36:29  
He.  
No.  
So there is there's many things happens on the other one pub site.  
So the data that we get from here is not that straightforward.  
Uh, so once they received this publishing kickoff time from AWE, we do bunch of processing in the name of injection where we use the one single GUID that AWE provides and then we have to retrieve all the relevant image informations are the media that SMF team created for each each file.

 **Dhiraj Singh** 36:54  
Hmm.

 **Aravind Siddoju** 37:05  
So we basically process that complete information try to.  
Store the details and if it has they aren't ohm provided in the right format, so there are places where we have actually need to copy these details to our BLOB storage and create stacks Uris for scan.  
I'll just talk in a minute on like different operations we do on the media side and why we have to process that.  
It's not that straightforward in a single answer that we don't just simply get this data and process it further.

 **Dhiraj Singh** 37:33  
I see.  
As he stood on a high level publishing kickoff is a go ahead, say Media ID for per channel.

 **Aravind Siddoju** 37:44  
Like.

 **Dhiraj Singh** 37:45  
We go and process it and one pub, in some cases copies, payload and other additional processing and there's some sort.

 **Aravind Siddoju** 37:50  
Right, right.  
Right, right.  
So yeah, again, if you just look at on the on the services perspective and or on the work plus perspective.  
So this is how it looks like.  
Uh, if you just look at some AWE to one pub.  
AWE just basically ohh cars are I publish I published service a comment to the one pub and from I publish it, it gets routed to different workflows that we have for the media depending upon as I mentioned.  
Like you do have a bunch of activities for the stages that we have are constructed on the AWE place.  
So we do need this these workflows that needs to be executed for each media publishing.  
So we we have workflows to create, ingest, publish, approved, cancel and go live.  
So which aligns with like different rings that we were published to eventually reach to customers during the B release.  
So basically the workflows A in the within them connect to the actual the media core services.  
So each workflow calls.  
It's the basic the the common services named media mediator service and the configuration service.  
So every logic app basically routes that request to these two.  
Actually, the media service endpoint and this guy takes responsibility of routing it to the actual.  
Core service depending upon the operation type and the update type that we get from the AWE request.  
So that's how eventually the the core media services gets kicked off and eventually once they are done, depending upon again the the business logic that we need to do they during the publish, they eventually called to the actual endpoint where we actually publish our media too.

 **Leslie Trowbridge** joined the meeting

 **Aravind Siddoju** 39:57  
And and in terms of like reporting back or a sending a signal back to AWE on how we finish.  
So basically, uh workflow calls the stateless service, which are basically the Web API that we expose and we and the workflows expect a callback URL and it's up to the it's on the media services to do a callback whenever they are finished and once it receives the callback from the actual media services, these workflows send a notification event to AWE telling hey I'm finished and that's how AWE kind of orchestration most of the next step next step and maybe it will.  
We can take care of any other manual or approvals it needs before it kicks off the next step, but basically it just goes to.  
I publish it eventually reaches our workflows which are logic apps and the logic apps internally calls a common service named mediators and configuration service.  
The Mediator service uses the configuration service and this configuration service will give the actual route to route the request to the actual media core service.  
So that's how it routes and throw the whole time the callback URL gets passed and whenever the operations are done the workflows expected callback URL from the core services depending upon whether the operation succeeds and fails.  
And that's when the workflow also gets finished.  
Fails data succeeded state and it eventually returns the successor failure event.  
Put some message in AWE.  
Service bus appreciably consumes, and that's how it kind of moves further on.  
The AWE side I see some hand raised.  
Any questions?

 **Mohit Gupta** 41:52  
I'll let Alexey go first.

 **Alexey Loginov** 41:56  
Thanks Margaret.  
Umm, just for me to understand those services.

 **Aravind Siddoju** 42:05  
Umm.

 **Alexey Loginov** 42:05  
Do they talk to each other?  
Aravind or communication done through logic app through logic up calls one service get something calls another service or services are allowed to call each other during the flow.

 **Aravind Siddoju** 42:20  
They talk to each other, Alexey, but not not the actual core services.  
So basically, ingestion doesn't talk to publishing.  
Publishing doesn't talk to us so well.  
It's not like that, but the services, it's not just one service.  
Which workflow calls and it just responds back, so it just goes through a multiple services eventually to transfer call back once things are done.

 **Alexey Loginov** 42:49  
Can you give an example of this?

 **Aravind Siddoju** 42:51  
OK.  
I'll.  
I'll just simply show shirt.

 **Alexey Loginov** 42:52  
This talk and and how is it rest call remoting call?

 **Aravind Siddoju** 42:57  
Right.  
I'll. I'll.  
I'll yeah, I'll simply show maybe how that goes.

 **Alexey Loginov** 43:00  
Umm.

 **Aravind Siddoju** 43:01  
That kind of picture is to open my studio.  
This is what I just want to show up but.  
Alex.

 **Mohit Gupta** 43:11  
And.

 **Aravind Siddoju** 43:19  
So this is our our, the actual media uh solution looks like and this is the the stateless service topic service that we use and we do have a bunch of stateless services which I just mentioned the media services in the the diagram and we do have this service fabric application which kinds of posts are these media services.  
We do have a bunch of libraries which does that.  
I just don't want to go in that detail.  
Maybe I'll just cover the basic flow on how the logic app actually features different services, so I just I just mentioned.  
So we do have this mediator service.  
We just exposes this controller.  
Like so, the logic app will basically call this controller.  
Umm.  
If you people can understand so it costs through the web hook step that we have on the workflows and it just routes that request to this mediator endpoint providing the operation name and it also provides this release info which we basically get from the AWE which contains the release information of what product payload, update type and release ticket information and so and so and then it also sends a callback URL the logic app while calling this API and we do have this request ID and bunch of few other details that it sent.  
So this guy just basically takes this request, goes further, goes further and it starts executing in this task factory, which happens asynchronously and the logic app basically waits for a callback.  
So it it's not just the return accepted.  
Umm, that's bound to something.  
Not it's not something that logic app awaits for.  
So basically it's a web hook where logic app sends the callback URL and it just listens to that callback URL and we are not going to send a call back URL right away.  
Rather, we start a different thread where we start operations for media and we immediately return the accepted state and this is where it again starts routing the actual breakfast.  
Which mediator service received?  
So the actual endpoint.  
So I'm not going to go line by line, so basically it reaches here.  
It uses a service name configuration service.  
The Mediator service, which contains a bunch of documents.  
I'm just gonna show you this.  
So just contains a bunch of, uh.  
Details on the route details for each update type, followed by the operation.  
So that's that's how the the media service will send the request to configuration service telling.  
Hey I wanted to get to a route for ingestion for the base Windows container update type.  
So this guy will give this entire URL the base address would be the the one pub endpoint as followed by this.  
So this will once it receives the request, immediately based control will just send this request to the actual endpoint that you get through this configuration discovery service. Umm.  
So let's say in this case like it's it's it's a media ingestion for base Windows container and this is the endpoint.  
So our service would be this media ingestion service public service, which is a common service for ingestion and it reaches, yeah, this matter.  
So this is the actual media core service which we wanted to reach and that's how the the the request reaches here from the logic app.  
So Logic app sends a request to the Mediator service Mediator service uses the configuration service to find the right route for update type and.  
Operation and eventually it gets that route using this JSON file and eventually it reaches the actual service.  
The core service.  
This could be anything here, but I'll just talk about the main services, so that's how it reaches.  
So ingestion it reaches here and this is where the ingestion part of media publishing happens.  
Did that answer your question Alexey?

 **Alexey Loginov** 47:59  
Well, yeah, at least at least gives some to follow up on.  
Thank you.  
Mm-hmm.  
Thank you.

 **Aravind Siddoju** 48:04  
Like so, yeah, again, the callback that I just mentioned, so some closely here is where we we do work callback once we start executing things and once we are done with whatever we wanted to do on the the ingestion operation of the media and once that's done that's where we we actually do this.  
The callback response to the the large CAP using the callback URL that we received in the 1st place from the logic app itself.  
Uh, yeah, that's how the media services will notify back to logic app telling whether they are done or not.  
And eventually the logic app had a little logic telling.  
Once this step is done, send a notification to AWE whether it's a success or failure.  
That's how logic apps will basically send a notification to the AWE to process further.  
Umm.  
And any other questions?

 **Mohit Gupta** 49:12  
I think our still has his hand up, but before before heading to him, Aravind, have a question for you.

 **Aravind Siddoju** 49:16  
OK.

 **Mohit Gupta** 49:18  
So does the logic app just call once and then let's the service fabric cluster do its magic with all the services or the IT doesn't do any orchestration right?

 **Aravind Siddoju** 49:19  
What?  
Like.

 **Mohit Gupta** 49:32  
It just calls it once.

 **Aravind Siddoju** 49:33  
So yeah, logic app doesn't do any.  
And smarting over, there basically calls the.  
Basically calls into the the stateless service of media to service.  
It expects for a callback.  
That's all it it does.  
It really doesn't know where to reach out to or anything, so pretty much all the logic apps basically called this media media service endpoint just deploy the operation type with the release info and it's media service.  
Who is responsible for routing it to the right media course service and eventually the media?  
The media services will do a call back once they are done.

 **Dhiraj Singh** 50:14  
So Ervin, in saying the ingest workflow, are there more logic apps on the right side where you have this business logic or that ingest logic app is the only one like I'm not able to clarify here.

 **Aravind Siddoju** 50:18  
Uh-huh.  
Right.  
So yeah, I yeah, I got your point.  
So yeah, again this is this is for different things.  
Ingestion doesn't do any further uh propagation on the forward side.  
So induction as I mentioned so Viraj you just asked, right, what we do with the data or the data processing, uh, storing the data in different formats.  
But those are all operations that we do in ingestion.  
Uh, ingestion, we we we don't proceed further, but it's a basic higher diagram, but the right side flows actually happens on the the publish operation part where we basically call the publishing core service which eventually uses the adapter service or the actual endpoints to reach out to the the the final customers to publish the media.  
Uh, that's the right cell propagation, so it's not necessarily all the core services.

 **Dhiraj Singh** 51:21  
So on the published logic app you have published a logic app on the left side.  
It causes API management.

 **Aravind Siddoju** 51:25  
Right.

 **Dhiraj Singh** 51:27  
Once you are in this orange zone, then you there are no logic apps involved.

 **Aravind Siddoju** 51:29  
Uh-huh, right.

 **Dhiraj Singh** 51:31  
Here you're saying I see.

 **Aravind Siddoju** 51:33  
No, no, but no.  
The logic app pretty much stays here itself, so there isn't any logic apps on the right.

 **Dhiraj Singh** 51:44  
I see.  
OK, this is yet, uh, different model than what, OK.

 **Aravind Siddoju** 51:46  
Like.  
Right, right.  
So yeah, 11 concern that we had in mind is, so we kind of web hooked every logic app to the the all the the media services.  
So at any point of time, if something fails and the the core media services ohh, eventually the logic app fails.  
So ideally it's should not be that directly tight coupled, so it's something that we are working on.  
But I'm just giving you here on that.  
And the other thing that I just wanna share is like although I just mentioned the core media services as these four are like four or five services ingestion creation, publish approval and Goel life.  
So not all these.  
So we don't have just only these four phase services.  
So few of the services are like common service for all update types and rest all are like individual stateless services which are for each update type I selected for each update type.  
I just wanna show that as well.  
So you just look into ours stateless services of the services that we had for the media.  
So although you can see only those four services are just described on the the bigger picture diagram.  
But if you just look at over here, only the well, the media ingestion service.  
Uh, and we do have media.  
Uh mediator service, which basically routes the information.  
Umm.  
And we do have this configuration service which are a few of the services and we have document manager services which are like pretty much common to all the update types.  
But if we look into the the publishing part of each update type, although it looks like a single service over here, but on a on the in reality we we do have individual stateless services for each update type.  
So we do have this audio file operation service which is which will take care of the the publishing part of.  
Sorry not audiophile operations of the autopilot shared service.  
Sorry, stateless service which is responsible for the publishing of APPF media and then we have of PAND sarissa big service which is actually the the core service for publishing the pan media.  
Similarly, we have this server container service fabric service which is again an isolated service which we use for publishing the container media.  
So other it looks like a single publishing service and the bigger picture.  
But yeah, these publishing services are pretty much 121 isolated for each update type, but that's one more thing. Uh.

 **Dhiraj Singh** 54:34  
Let's see.

 **Aravind Siddoju** 54:38  
And what I want to cover.

 **Mohit Gupta** 54:42  
Arvind uh actually has a has his hand up for for a long time and almost and we are almost on time.

 **Aravind Siddoju** 54:44  
Umm.  
Good achal.

 **Achal Shah** 54:47  
No. Yeah.

 **Aravind Siddoju** 54:48  
Sorry, go ahead.  
I'm not seeing any hands.  
Sorry my speaking not showing up.

 **Achal Shah** 54:52  
Yep.

 **Mohit Gupta** 54:53  
So we'll have to maybe leave it for next time.

 **Aravind Siddoju** 54:57  
So no problem.

 **Achal Shah** 54:57  
Yeah, no.

 **Mohit Gupta** 54:58  
That'll go ahead with your.

 **Aravind Siddoju** 54:58  
I shall go ahead.

 **Achal Shah** 54:59  
Yeah.  
And maybe we can talk about this further later, but I noticed that when the mediator service calls any of the other services, it it again uses or it's a rest call, right?

 **Aravind Siddoju** 55:08  
Mm-hmm.  
Right.

 **Achal Shah** 55:12  
Even though all the services live in one service fabric cluster, right?  
So is there any service fabric remoting or is it everything happens through rest walls between the services?

 **Aravind Siddoju** 55:17  
Right.  
A pretty much everything is auto rest call, although we do use couple of services for remoting.

 **Achal Shah** 55:29  
OK.

 **Aravind Siddoju** 55:32  
Although it's not the the main main course, services are like hooked up with rest culture itself, so we do have a different service name.  
This audio file operation service she which we created to interact anything with the audio pipelines but triggering any published but we we kind of connected through Remoting clients but all the main publishing and the main media course which is are like tied up with the rest API calls itself.

 **Achal Shah** 56:01  
OK.  
Yeah. Thanks.

 **Mohit Gupta** 56:08  
I've been sorry.

 **Aravind Siddoju** 56:08  
Yeah, that's sure.

 **Mohit Gupta** 56:08  
We'll have to.  
We'll have to stop here, in the interest of time.

 **Aravind Siddoju** 56:10  
No problem.

 **Mohit Gupta** 56:12  
I was hoping to get through to at least one update type today, but I think you have to.

 **Sam Son** left the meeting

 **Mohit Gupta** 56:17  
Thanks.  
So you all the steps are we will resume this with any questions in general about the high level architecture and design and then we will maybe identify a set of incidents.  
We'll go over one type of incident, one class of incidents and maybe we can devote 15 minutes each for each class.  
And then try to identify any issues in the system.

 **Dhiraj Singh** 56:46  
Yeah, this was great, by the way.  
I think I understood the architecture for the first time at every high level and the way I understand is these services are kind of set up in a microservices model and the logic app is really responsible for holding.

 **Leslie Trowbridge** left the meeting

 **Dhiraj Singh** 57:02  
The the long running task and interfacing with AWE and interactions are happening here.  
So this is a different model than some other pipelines, but this is great to know now.

 **Aravind Siddoju** 57:16  
Yeah.

 **Mohit Gupta** 57:19  
Cool.  
So I will, I will.  
Uh.  
Maybe publish an agenda for next meeting in advance and we will have better preparation next time.  
This was just one day notice, so apologies if if things were not as per your expectations, but we'll be better next time.

 **Dhiraj Singh** 57:36  
This was great.  
Awesome.  
Thank you very much.

 **Mohit Gupta** 57:41  
OK.  
Thank you all. Once again.

 **Vaithi Krishnasamy** left the meeting

 **Mohit Gupta** 57:44  
See you next time.

 **Alexey Loginov** 57:44  
Excellent.

 **Aravind Siddoju** 57:44  
Thanks everyone.

 **Alexey Loginov** 57:45  
Typical.  
Thank you.  
Thank you. Bye.

 **Achal Shah** 57:47  
Bye.

 **Mohit Gupta** 57:48  
Thank you. Bye.

 **Alexey Loginov** left the meeting

 **Antonio McMichael** left the meeting

 **Achal Shah** left the meeting

 **Fabiola Ansara** left the meeting

 **Dhiraj Singh** left the meeting

 **Bhaskar Verma** left the meeting

 **Jacob Kissel** left the meeting

 **Dinesh Vijayakumar** left the meeting

 **Aravind Siddoju** left the meeting

 **Richard Jimenez** left the meeting

 **Yashasvini Rathore** left the meeting

 **Fatuma Omer** left the meeting

 **Deepak Kunwar** left the meeting

 **Mohit Gupta** stopped transcription