penthera | Video Insights

penthera

Understanding and Solving Monetization Problems for SSAI/DAI in VOD

By: Scott Halpert, SVP Product and Partnerships

Q3 2022

penthera.com

INTRODUCTION

This whitepaper describes the process, benefits, and limitations of Server-Side Ad Insertion (SSAI). It also introduces Penthera's 2nd TM Look product which addresses limitations of SSAI to drive higher fill rates, render rates, and CPMs for AVOD publishers.

WHY USE SERVER-SIDE AD INSERTION

Historically, ad insertion for both web/mobile display ads and video ads was handled by "client-side" technologies. That is not to suggest that there are no servers involved, rather that there is a direct relationship between the "client" (or end user's device) and the ad server making the ad decisions. This is an effective and efficient way to deliver advertising to "traditional" digital viewers. However, with streaming video, employing a client-side process, called Client Side Ad Insertion (CSAI) presents a number of challenges resulting in a poor user experience.

Delivering CSAI advertising within a video stream requires that the client effectively have two video players, one for playing the main content asset and the second for playing the ads. At an ad break, the content stream is paused while the ad player spools up and plays out the ad from the ad server. This handoff introduces latency (perceived as lag by the end-user) two times for each ad break: during the transition from content to ads, then back again to content. Additionally, differences in encoding and formats between content and ads can cause playback to stall or "break" after each handoff, potentially resulting in a terminal error that ends the stream. Finally, CSAI is vulnerable to being blocked by ad blockers running on the user's device.

SSAI was developed to address these problems with the goal of enabling a viewing experience similar to what users are accustomed to on broadcast TV. While technically more complex than CSAI, it offers a number of distinct advantages. Key amongst them is the ability to deliver a single stream that includes both the main content asset and the advertising. Transitioning between the content and advertising elements is "seamless", more stable, and ultimately delivers a superior viewing experience. Delivering ads via SSAI supports the same granular targeting as client-based technologies, while eliminating the risk of ad blockers.

With SSAI, the advertising for a stream must first be segmented and encoded into a HLS or DASH format so that it can be played by the player. This task is handled by a technical service called an ad stitcher (the 'server' in server-side ad insertion). When the ad stitcher receives an ad, it will encode the ad asset such that it becomes compliant with the encoding/segmentation protocols of that publisher. Once encoded, it is ready for insertion into the stream at the stitcher, ensuring that the ad is part of the single continuous stream received by the player. By taking this approach, SSAI enables publishers to eliminate the problems described above with CSAI.

HOW SSAI/DAI WORKS FOR VOD

Streaming video is typically delivered using a protocol called Adaptive Bit Rate (ABR). ABR streams are broken into individual segments, typically 5-6 seconds of video per segment. Segments are encoded in different qualities (high to low) so the player is able to make frequent decisions on which quality level of video to request based on the available network bandwidth. This ensures a continuous stream at the highest possible quality. The player uses a file called a manifest to know the location (e.g. server) to retrieve any given segment from. The manifest is the index for all segments of the video. If the viewer skips to 1:15:30 into a stream, the manifest would tell the player the location of the appropriate segment for the available bandwidth (e.g. #755, highest bit rate).

When streaming live or linear content, the player will request incremental manifests from the origination server. For example: every few minutes of content will be its own manifest with triggers inserted to initiate ad decisions within these "short" manifests. Because of this process of continually handing manifests to the player, ad insertion in live and linear can be handled much like all other digital advertising. That is to say it is managed on a "just in time" basis, where the ad decisions for the stream are triggered usually within thirty seconds of each ad break and on an individual ad break basis.

For VOD, this process is meaningfully different, and different from any other process in digital media. When the viewer presses play, the player requests and receives the full video manifest up front, before the video begins. The player only receives the manifest this one time. This approach enables key user facing functionality. Users can easily "scrub" back and forth through the video and are presented with a timeline so they can see how much of the video they've watched and how much of the video remains.

However, since the player will only ask for the manifest one time, the ad decisioning process is changed substantially. All of the ad decisions for the entire stream must be made in the time between when the user presses "play" and when the first frame of the video renders on their device.

Since OTT/CTV publishers don't want users waiting for an extended period of time before their video starts, this requires the ad decisioning and insertion process to happen quickly. It's not a simple process. Multiple interactions must occur between ad servers, the ad stitcher, and programmatic systems, each involving numerous network hops to accomplish. As such, there are tight timeout limits placed on each element in the process to ensure that the video start-up delay is as short as possible for the user. The tradeoff between decision complexity and user experience drives a significant amount of failed or under-monetized ad placement opportunities. Figure 1 shows a high level architecture and interactions of video delivery with server-side ad insertion:

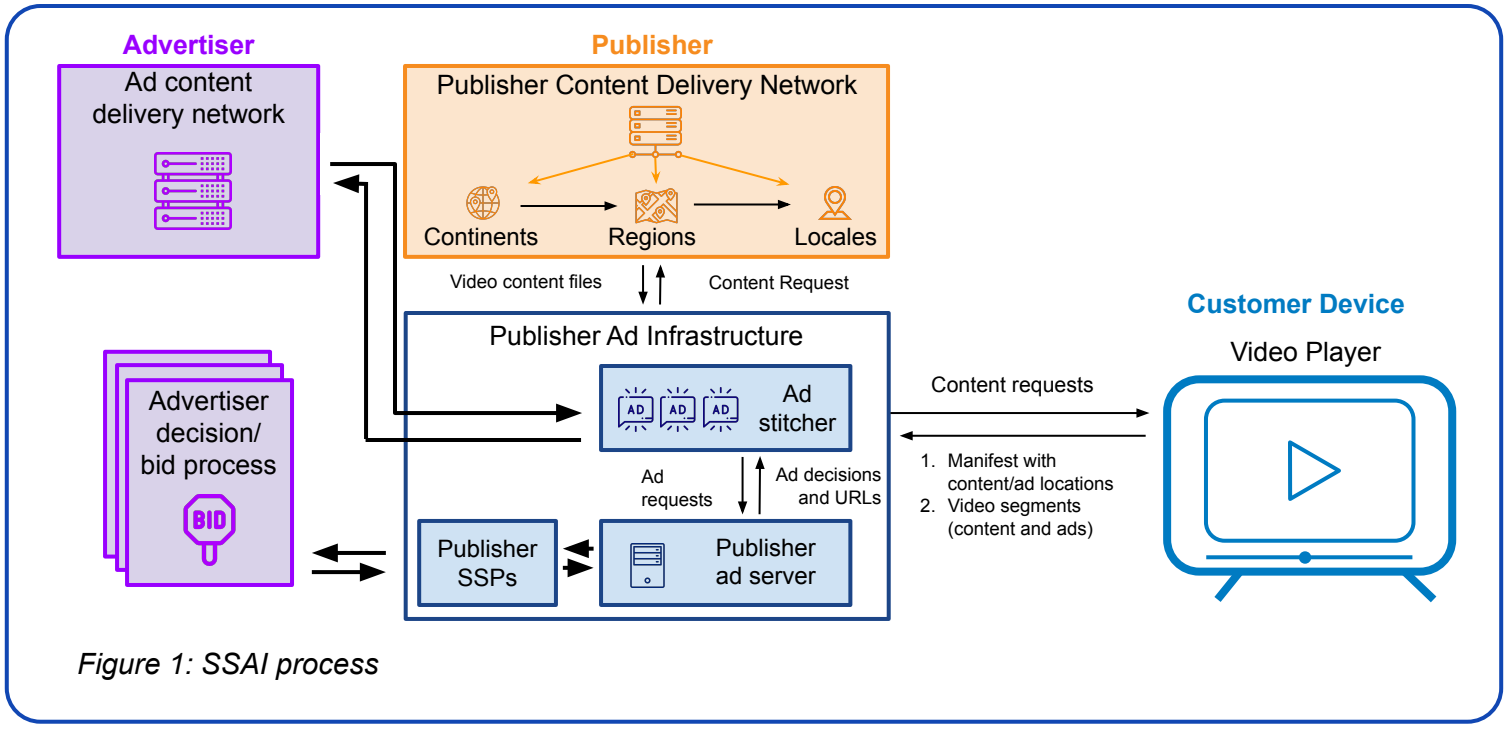

Figure 1: SSAI process

Here's an example of the steps that occur in a typical SSAI process where the publisher has both direct sold and programmatic demand:

1. User presses "play" triggering the player to call the ad stitcher, passing the asset information and any available targeting information,
2. The ad stitcher receives the request, identifies the requested asset, pulls it from the origination server,
3. The ad stitcher interrogates the asset to determine the number and duration of the ad breaks in the asset:
    a. Note: the ad breaks and their target durations (i.e. 4 ad pods, 2 min each) are typically pre-determined by the publisher
    b. These ad breaks are identified with markers to indicate their location and duration
4. The ad stitcher transmits the ad break information and associated targeting data to the publisher's ad server
5. The ad server looks for "internal" campaigns and may simultaneously call one or more Supply Side Platforms (SSPs) for "external" campaigns:
    a. The SSPs will offer the advertising breaks to Demand Side Platforms (DSPs)
    b. The DSPs will respond with 0 or more "bids" to the SSPs
    c. The SSPs will decide which advertiser(s) "win" the opportunity to purchase the available ad spots
    d. The SSPs transmit their results to the publisher's ad server
6. The publisher's ad server determines which advertisers will make it into the stream and in what positions within the stream they will be placed:
    a. Note: determining position is variable in terms of flexibility. Oftentimes advertisers will "bid" on specific pods or placement within those pods (i.e., pod 1, 1st position). In this case, the publisher's ad server cannot move these advertisers to another spot in the stream
7. The final ad load decisions are transmitted to the ad stitcher
8. The ad stitcher fetches the pre-encoded ads and "stitches" them into the stream by inputting their locations into the manifest:
    a. Note: if there is a new ad (i.e. one not seen before for that publisher) the ad stitcher must first download and encode the ad. Oftentimes this means the new ad will not be inserted into the first X number of streams to which it was targeted
9. The ad stitcher hands the, now complete, manifest to the player
10. Playback begins

For user experience purposes, this entire process must be completed within 2-3 seconds, preferably faster. This is a complex process, where numerous 3rd party interactions are required and problems can occur.

PROBLEMS RESULTING FROM SSAI/DAI WITH VOD

As we describe above in VOD with SSAI, all of the ad decisions are made between the time the user presses "play" on their device and the first frame of the video rendering. This is because they must be "stitched" into the stream (technically the stream's manifest) before playback begins. Offering the stream's associated inventory simultaneously and well ahead of when the ads are going to run has negative effects on:

Render Rate: Render Rate (the number of ads viewed divided by the total number of ads inserted into the stream's manifest) is negatively impacted by the process. Since all of the decisions are made up front at the beginning of the process, any time a user leaves a stream prior to completion, any ad that is scheduled for after that point in the stream will not "render". It is common for render rates on VOD to be 50%-60% or lower. Low render rates are taken into account by DSPs and they adjust their bidding/pricing accordingly. This results in lower advertiser participation and reduced average CPM and fill.

Publisher's Ad Server: Every campaign on an ad server has budget and pacing constraints. The ad server's job is to keep that campaign on the ideal curve that satisfies all of the campaign's constraints. By asking the ad server to decide at a point in time well ahead of when the ad is going to run, it is forced to constrict the pool of eligible campaigns. As a result it is choosing from fewer campaigns for any given VOD stream. This reduction in the pool of advertisers has a negative effect on fill and average CPM (at both the stream and pod level).

Programmatic Systems: Programmatic buying systems (DSPs) are exceptionally capable of making individual decisions at scale. These decisions are influenced by multiple factors that include: budget/pacing, information about the user/device/content/etc., and how the advertiser has interacted with that user in the past (e.g. did they just show an ad to them 5 min ago) and the likelihood of that user performing the desired action after seeing the ad (e.g. click, brand recognition, buying something, etc.). Programmatic systems are not good at looking at multiple ad opportunities that relate to the same user simultaneously, which is exactly what happens in a VOD stream with SSAI. For example, if the stream has 4 pods, the DSP won't know if it won the 1st pod before having to bid on the 2nd and so on. Since the pods are offered simultaneously, the DSP needs to make a decision for all of them at the same time. As a result, they change their bidding behavior. This results in uneven responses/bidding across the pods. While there may be enough advertisers to completely fill all of the pods, because of the way they have bid (i.e. 1st pod, 1st slot in 1st pod) they can't be reallocated to other pods in the stream. This results in over participation (over fill) in some pods and under participation (under fill) in others. Once again, this drives down fill, yield and average CPM on a per pod basis.

Timeout Windows: There are multiple timeout windows that come into play in the ad insertion process for SSAI and all are impacted. Because the user is waiting for the stream to start while the ad decisioning process is running, the timeout windows at the SSPs and ad stitcher are set to be relatively short. The timeout at the SSP, if expanded, would generally yield more advertisers bidding. However this can't happen in the current process. Ad stitchers also encounter problems when they see a new ad for the first time because they need to encode that ad before inserting it into a stream. Beyond these, each advertisement has a Time-To-Live (TTL) associated with it. TTL is a requirement that the ad server receive confirmation within a certain period of time that the ad was presented to a viewer. Typically, these are set for 3-6 minutes for all digital delivery. However with VOD, where the first ad pod might not run for 8+ minutes, this becomes problematic. As such, specific campaign line items for VOD need to be set up with much longer TTL windows. This creates a failure point where, by human error or technical problem, these windows are set inappropriately. The publisher could choose the ad correctly, have it inserted into a stream and delivered, only to find out that the advertiser has invalidated the delivery due to a TTL violation. This causes friction and lost revenue for the publisher. Figure 2 shows the SSAI process and highlights the typical problems that occur:

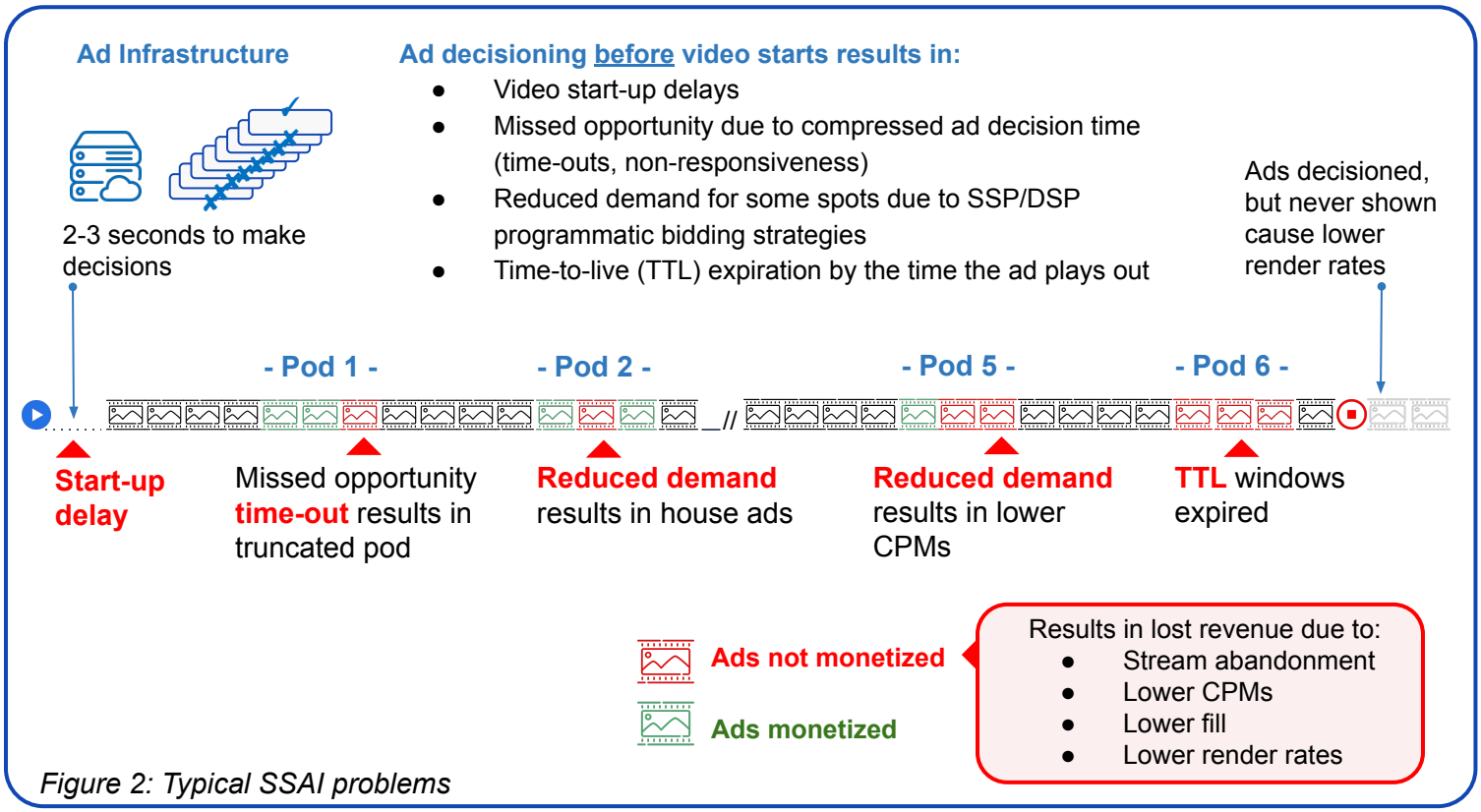

Figure 2: Typical SSAI problems

A SOLUTION: 2nd LOOK

2nd Look is a cloud service that solves these critical limitations of SSAI and drives increased render rates (# ads played/# ads decisioned and inserted into the stream), fill, CPM and overall revenue. It accomplishes this by time shifting when the ad decisions need to be made. In doing so, 2nd Look enables the publisher's ad server and programmatic systems to operate more efficiently. It does not replace any part of the publisher's ad stack and is interoperable with any VOD stream, from any type of device that uses SSAI (e.g., CTV, mobile, PC).

HOW 2nd LOOK WORKS

2nd Look acts as an intermediary between the player and the ad stitcher. In this position, 2nd Look is able to alter the typical manifest playback process so that ad segments can be changed after playback begins. With 2nd Look implemented, the manifest requests ad segments from the 2nd Look service which works with the ad stitcher to decision and locate ad assets throughout the stream. This affords publishers the ability to make decisions about what those ads should be after playback begins. Figure 3 shows how 2nd Look is implemented within a typical server side ad insertion architecture:

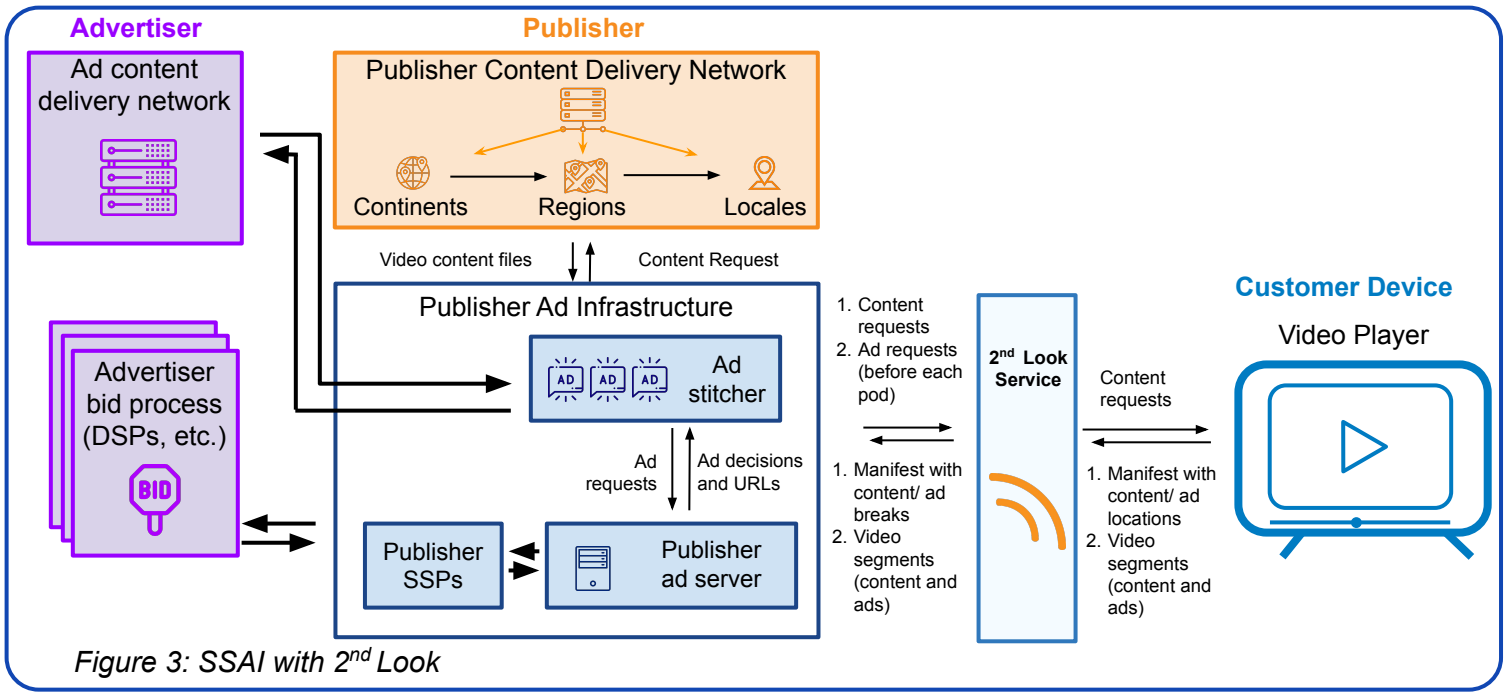

Figure 3: SSAI with 2nd Look

2nd Look changes the workflow for VOD with SSAI in a subtle, yet impactful way. An example of the ad insertion process, with 2nd Look in place, works in the following way:

1. User presses “play” triggering the player to call 2nd Look passing the asset information and any available targeting information
2. 2nd Look receives the request, identifies the requested asset, pulls it from the origination server:
    a. If a pre-roll ad is required, 2nd Look will, via the ad stitcher, trigger the decisioning for that ad
3. 2nd Look returns the stream manifest to the player
4. Playback begins
5. As the player approaches each ad break, it makes a request to the 2nd Look service which, in turn, transmits to the ad stitcher a request for the currently approaching ad break (i.e. request for 2 min of ads for pod #1)
6. The ad stitcher transmits the single pod request and associated targeting data to the publisher’s ad server
7. The ad server will look for “internal” campaigns and may simultaneously call one or more Supply Side Platforms (SSPs) for “external” campaigns:
    a. The SSPs will offer the advertising breaks to Demand Side Platforms (DSPs)
    b. The DSPs will respond with 0 or more “bids” to the SSPs
    c. The SSPs will decide which advertiser(s) “win” the opportunity to purchase the available ad spots for that pod
    d. The SSPs transmit their results to the publisher’s ad server
8. The publisher’s ad server will determine which advertisers will make it into the pod and in what positions within the pod they will be placed
9. The final ad decisions are transmitted to the ad stitcher
10. The ad stitcher fetches the pre-encoded ads
11. The ad stitcher hands the, ads to 2nd Look
12. 2nd Look informs the player about the locations for the ads in the pod and they are played out

2nd Look is designed in such a way that the key features of VOD viewing and SSAI insertion are maintained. For example, users can still scrub within the video and see the overall video timeline, while publishers can still deliver advertising as described above within the SSAI approach. Furthermore, 2nd Look is able to truncate ad pods such that the duration of the pod closely matches the duration of the ads being inserted. This is important for scenarios where the ad pod is not completely filled by the ad decisioning systems (i.e. 2 min ad pod is filled with 1:45 min of ads). Figure 4 illustrates how 2nd Look corrects the problems associated with SSAI.

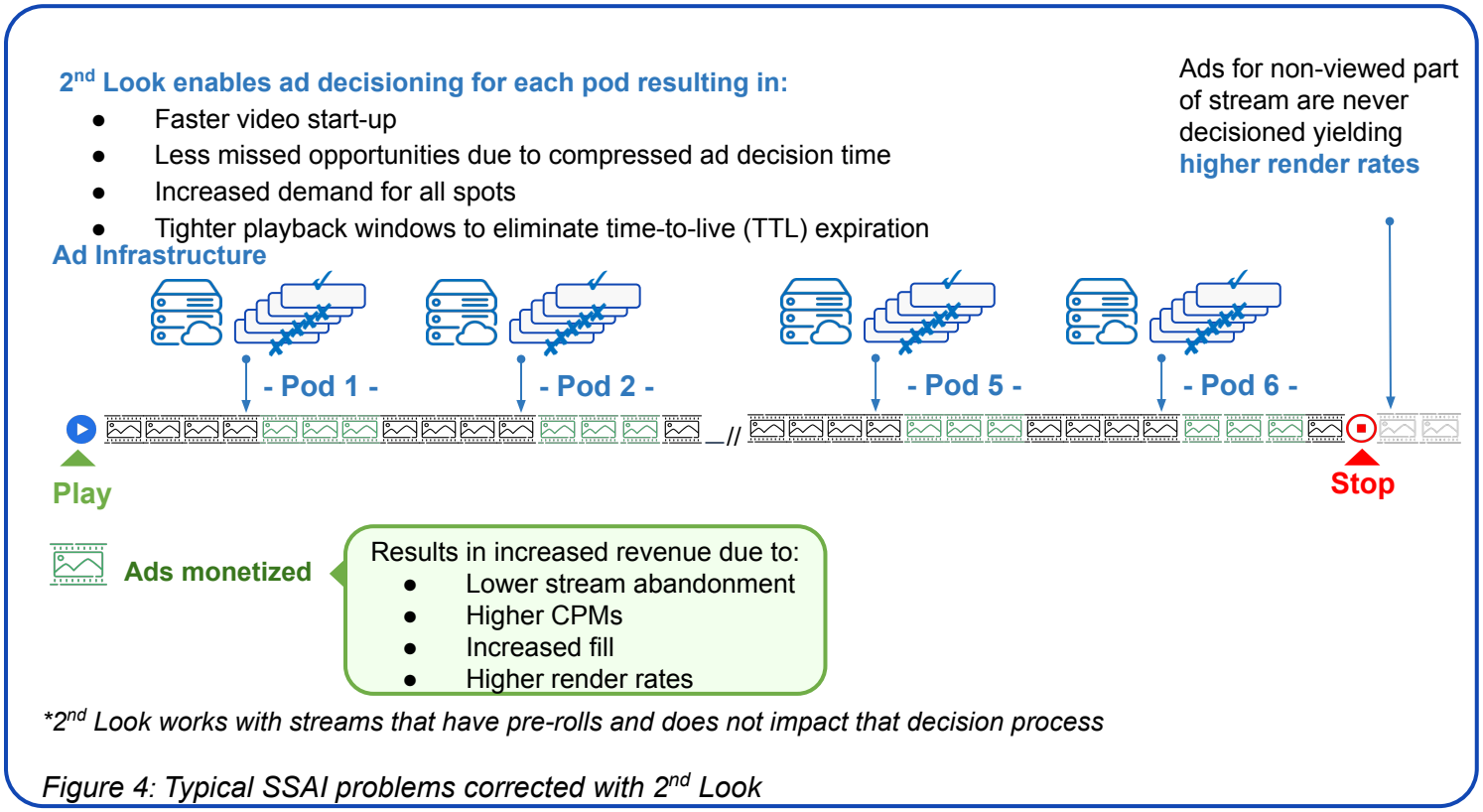

Figure 4: Typical SSAI problems corrected with 2nd Look

HOW 2nd LOOK SOLVES SSAI PROBLEMS FOR VOD

By altering the timing of this process and calling the ads just before they are going to be played by the player, 2nd Look changes the dynamics described above in meaningful ways.

Render Rates: Triggering the ad decisioning just prior to when the ads will run will substantially increase Render Rates. With 2nd Look, ads that are called are very likely to be played. If the user exits the stream, ads beyond the point of exit are likely to not have been inserted yet. This will, over time, affect the DSPs' bidding behavior since they will no longer "discount" bidding frequency/price due to low Render Rates.

Publisher Ad Server: The publisher's ad server will be able to have greater advertiser participation across the stream since ad decisions will be made much closer to when the ads will run and in a serial process. As such, an advertiser who was not able to buy "right now" for pod 1 may be able to buy pod 2 when it is offered a few minutes from "now".

Programmatic Systems: Offering the pods in serial will enable DSPs to change their bidding behavior. For example, assume DSP 1 has advertisers A, B and C that want to buy this user/stream session, but who don't win a slot in pod 1. When pod 2 is offered, DSP 1 will still have an opportunity for advertisers A, B and C to bid and win a place in that pod or subsequent pods. This increases advertiser participation in the inventory offering, driving up fill and average CPM through competition. Competition does not necessarily imply variable price bidding (since most prices are pre-negotiated), rather increased participation, on average, reduces the price differential between the top advertisers and therefore overall pod CPM.

Timeout Windows: 2nd Look can control the timing of when ad decisions are triggered, maintaining a short enough elapsed time (ad decision to ad view) to insure TTL windows won't be violated while at the same time offer more time for programmatic partners and the ad stitcher to get more buyers and encode new ads, respectively. Doing so will result in more advertiser participation and provide ad stitchers with time to encode and insert new ad creatives into the stream.

2nd Look can be easily integrated with a publisher's player via simple HTTP interfaces and Penthera will work with the publisher's ad stitcher (via their standard APIs) to integrate and test the system before launch and scaling.

About Penthera

Penthera is a global software company solving video streaming and monetization issues for CTV and OTT publishers. Its AVOD monetization solution provides flexibility to make better ad decisions resulting in increased fill, render rate and CPMs. Penthera's technology is currently deployed in 36 countries across 6 continents.

Contact

Scott Halpert
SVP Product & Partnerships
scott.halpert@penthera.com

Connect with Penthera

Website: penthera.com
LinkedIn: / Penthera/
Twitter: @WeArePenthera